REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE 17 April 95

3. REPORT TYPE AND DATES COVERED Final 9/1/93 - 3/10/95

4. TITLE AND SUBTITLE Efficient Time Domain Solutions of Maxwell's Equations for Aerospace Systems

5. FUNDING NUMBERS F49620-93-C-0066

6. AUTHOR(S) Vijaya Shankar, W. Hall, G. Rowell, A. Mohammadian

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Rockwell Int'l Science Ctr Thousand Oaks CA

8. PERFORMING ORGANIZATION REPORT NUMBER AFOSR-TR 95-0341

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
DIRECTORATE OF AEROSPACE SCIENCES
BOLLING AFB, DC 20332-6448

NA

10. SPONSORING/MONITORING AGENCY REPORT NUMBER 2307/AS

DTIC
ELECTE
MAY 1 5 1995
S C D

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE
DISTRIBUTION IS UNLIMITED

12b. DISTRIBUTION CODE

19950511 105

13. ABSTRACT (Maximum 200 words)

This report contains the development of an unstructured grid–based finite–volume integration scheme for solving the time–domain Maxwell's equations to study a myriad of problems in electromagnetics. The principal application of this technology is the prediction of radar cross section (RCS) of low observable platforms. This work was performed under the AFOSR contract F49620–93–C–0066.

DTIC QUALITY INSPECTED 5

14. SUBJECT TERMS Maxwell Equations, time domain, unstructured Grid

15. NUMBER OF PAGES 170

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED

18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED

19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED

20. LIMITATION OF ABSTRACT U

Table of Contents

SUMMARY 1
1.0 INTRODUCTION 2
1.1 Attributes of CFD 3
1.2 Objectives 4
1.3 Computational Electromagnetics (CEM) 4
1.4 CEM Issues 5
2.0 MAXWELL'S EQUATIONS 8
3.0 GENERAL CONSERVATION–LAW FORM 11
4.0 CONSERVATION FORM FOR MAXWELL'S EQUATIONS 16
4.1 Interface Flux Forms 17
4.2 Characteristics of Maxwell's Equations 18
4.3 The Riemann Solver: Preliminaries 20
4.4 Discretization and Dissipation 22
5.0 UNSTRUCTURED FINITE–VOLUME TREATMENT 27
5.1 Space/Time Discretization 27
5.2 Polynomial Representation and Least Squares 28
5.3 The Unstructured Second–Order Algorithm 29
6.0 UNSTRUCTURED GRIDDING 31
7.0 MASSIVELY PARALLEL COMPUTING 40
8.0 FURTHER DEVELOPMENT OF UNSTRUCTURED–GRID CEM CODE . . 44
9.0 REFERENCES 46
APPENDICES
A1. Treatment of Impedance Layer 49
A2. Treatment of Anisotropic Media 55
A3. Geometry and Gridding Techniques 57

Accesion For
NTIS CRA&I ☑
DTIC TAB ☐
Unannounced ☐
Justification
By
Distribution /
Availability Codes
Dist | Avail and / or Special
A-1

1.0 SUMMARY

This report contains the development of an unstructured grid–based finite–volume integration scheme for solving the time–domain Maxwell's equations to study a myriad of problems in electromagnetics. The principal application of this technology is the prediction of radar cross section (RCS) of low observable platforms. This work was performed under the AFOSR contract F49620–93–C–0066.

1.0 INTRODUCTION

The design and development of modern aerospace configurations increasingly require better understanding, management and exploitation of ever more complex physical phenomena in different disciplines such as fluid dynamics, structural mechanics, propulsion, controls, and electromagnetics that all play an interdisciplinary role. The traditional approach of serial design by individual disciplines which cannot account for coupling effects will not only result in poor design but also is not cost effective. Extensive simulations of the various complex phenomena are required to understand the interdisciplinary role in design. Along with advances in experimental simulations, with the emergence of the supercomputers, both conventional (vector Cray-like architectures) and massively parallel, the computing technology is beginning to play an ever increasing role in design simulations. Both government and industry view 'Computational Sciences', a discipline that exploits advances in numerical algorithms development and the increasing power of supercomputers, super-minicomputers and graphics workstations, as a critical, potentially efficient and cost-effective technology for advanced design.

The development of a computational environment that encompasses different disciplines for multidisciplinary studies as described in Fig. 1 requires that the following computational capabilities be properly matured within each discipline.

1) Computational Fluid Dynamics (CFD)

   — Turbulence modeling
   — Transition
   — Finite-rate chemistry
   — Algorithms for incompressible to hypersonic Mach number range
   — Unsteady and separated flows

2) Computational Electromagnetics (CEM)

   — Proper implementation of Maxwell's equations with general material properties
   — Formulation and implementation of different boundary conditions

3) Computational Structural Mechanics (CSM)

   — Accurate finite-element models for static and dynamic flexible effects
   — Failure and fatigue analysis accounting for material grain structure

The development of computational capabilities in all these disciplines critically depends on the following technologies.

1) Geometry and Grid Setup

   — Three-dimensional geometry modeling
   — Structured and unstructured grid cell representation
   — Pre and post processing for visualization

2) Computer Architectures

— Vector/parallel coarse grain machines such as the Cray C-90 with 16 CPUs
— Massively parallel machines, both SIMD (Single Instruction Multiple Data) and MIMD (Multiple Instruction Multiple Data)
— Advanced graphics workstations

One of the computational disciplines that has been a supercomputing pace setter for the last three decades is CFD. This technology, which started with the development of the transonic small-disturbance theory in the late 60's, has matured both in algorithm and code development to the point of today being able to solve the time averaged Navier-Stokes equations for predicting the flowfield over a complete fighter. Today, CFD is playing a critical role in the development of next generation fighters and the National Aerospace Plane, though only in single discipline mode. The development of a multidisciplinary computational environment can significantly benefit by applying many of the attributes of CFD to other disciplines such as CEM.

1.1 Attributes of CFD

1) The fluid dynamic equations are usually cast in conservation form either as differential or local integral equations conserving mass, momentum, and energy fluxes, thus allowing for numerical capture of flow discontinuities such as shocks and slip surfaces. Equations representing the physics in other disciplines, for example Maxwell's equations in electromagnetics, can be cast in similar conservation forms for conservation of appropiate fluxes.

2) Recent developments of hyperbolic algorithms for solving the time-domain Euler equations are based on the characteristic theory of signal propagation and are referred to as the 'upwind' schemes. For hyperbolic equations, the upwind based schemes can be constructed to provide the right amount of numerical dissipation to achieve both stability and accuracy. Current state of the art in numerical algorithms is based on Essentially Non Oscillatory (ENO) interpolation schemes that can provide arbitrarily high order of accuracy for arbitrary cell shapes such as hexahedral, triangular prism, and tetrahedral elements.

3) For treatment of complex aerospace configurations, CFD methods usually employ either a structured grid based body-fitted coordinate system or an unstructured finite-element grid setup for ease in implementing the various boundary conditions. Similar numerical geometry and grid setup procedures are equally applicable to modeling complex problems in other disciplines.

4) Pre and post processing capabilities running on advanced graphics work stations are effectively employed to visualize and animate the geometry/grid and solution.

The goal of 'Computational Sciences' is to effectively employ the many advances in CFD coupled with emerging supercomputer architectures with expected teraflops (trillion floating point operations per second) performance to mature the computational technology

in different disciplines and be able to perform multidisciplinary studies critical to advanced design.

1.2 Objectives

Toward establishing a computational environment for performing multidisciplinary studies, the initial goal is to advance the state-of-the-art in CEM with the following specific objectives.

1) Apply algorithmic advances in CFD to solve Maxwell's equations in general form to study scattering (radar cross section), radiation (antenna), and a variety of eletromagnetic environmental (electromagnetic compatibility, shielding, and interference) problems of interest to both the defense and commercial community.

2) Mature the CEM technology to the point of being able to perform coupled CFD/CEM optimization design studies.

3) Establish the viability of MIMD massively parallel architectures for tackling large scale problems not amenable to present day supercomputers.

1.3 Computational Electromagnetics (CEM)

The ability to predict radar return from complex structures with layered material media over a wide frequency range (100 MHz to 20 GHz) is a critical technology need for the development of stealth aerospace configurations. Traditionally, radar cross section (RCS) calculations have employed one of two methods: high frequency asymptotics, which treats scattering and diffraction as local phenomena; or solution of an integral equation (in the frequency domain) for radiating sources on (or inside) the scattering body, which couples all parts of the body through a multiple scattering process. A third approach is the direct integration of the differential or integral form of Maxwell's equations in the time-domain.

The time–domain Maxwell's equations represent a more general form than the frequency–domain vector Helmholtz equations, which are usually employed in solving scattering problems. A time–domain approach can, for instance, handle continuous wave (single frequency) as well as a single pulse (broadband frequency) transient response. Frequency–domain–based methods usually provide the RCS response for all angles of incidence at a single frequency, while time–domain–based methods provide solutions for many frequencies from a single transient calculation. Also, in a time–domain approach, one can consider time–varying material properties for treatment of active surfaces. By using Fourier transforms, the time–domain transient solutions can be processed to provide the frequency–domain response. Frequency–dependent (dispersive) and anisotropic material properties can also be included within the time–domain formulation.

CEM is a critical technology for the United States to maintain its leadership in the advancement of future aerospace development through supercomputing. As we transition from the present Gigaflops to the next generation Teraflops computing, CEM will become

integral to aerospace design not only as a stand alone technology but also as part of the multidisciplinary coupling that leads to well optimized designs.

1.4 CEM Issues

Proper development of a CEM capability appropriate for all aspects of aerospace design must consider various issues associated with electromagnetics. Some of them are:

1. Physics of Maxwell's equations

- differential and integral forms of Maxwell's equations suitable for numerically satisfying tangential field flux conservation
- flexibility in implementing total field, scattered field, and diffracted field forms depending on the nature of the problem being solved
- incorporation of various source terms

2. Material properties

- perfectly conducting surfaces
- lossy/lossless ϵ and μ
- resistive sheet (conductivity σ and thickness d)
- impedance layer
- anisotropic media
- chiral media
- dispersive media – $\epsilon(\omega)$ and $\mu(\omega)$
- nonlinear materials

3. Boundary conditions

- perfectly conducting, $n \times \mathcal{E} = 0$ (Electric wall) and $n \times \mathcal{H} = 0$ (magnetic wall)
  — accurate evaluation of $n \times \mathcal{H}$ on the PC surface is crucial
- material interface, $|n \times \mathcal{E}|$ and $|n \times \mathcal{H}|$ are zero
  — algorithms must account for any variation in ϵ and μ at interface
- resistive sheet, $n \times \mathcal{E}$ and $n \times \mathcal{H}$ jump across RS
- impedance layer, $n \times n \times \mathcal{E} = -\eta\, n \times \mathcal{H}$
  — implementation in time–domain involves convolution integrals
- nonreflecting farfield
  — characteristics based hierarchy of absorbing conditions for total field, scattered field, and diffracted field formulations
- periodic
- zero flux (collapsing cell faces)

4. Algorithmic issues

- unstructured grid–based finite–volume algorithms that include structured grids as a special case
- stability and accuracy of schemes using spectral techniques
- construction of higher–order schemes including at boundaries using polynomial representations for electric and magnetic field variations inside a general polyhedron cell
- implicit and explicit schemes with approprite space and time discretization

5. Gridding

- geometry modeling using CAD packages
- structured or unstructured surface grids using advancing front technique
- volume grids using hexahedral, prismatic or tetrahedral cells
- optimization routines for achieving grid quality and smoothness
- adaptive gridding

6. Massively parallel computing issues

- domain decomposition and load balancing
- internodal message passing with minimum communication delays
- synchronization for time accurate computation
- measure of MFLOP rating
- scalability measures
- pre and post processing in parallel environment
- transportability

7. User Issues

- problem set up and boundary conditions – how flexible and general is the code?
- geometry and grid set up time, resolution requirements and computational domain size
- internal consistency check for spotting user errors and dignostic measures
- run enviroment
  - — selection of number of nodes
  - — load balancing and domain decomposition
  - — automatic termination criteria
  - — complete monostatic runs
  - — post processing routines – FFT, plotting,$\cdots$
- Reliability of solution

8. Validation and Applications

- code validation on Electromagnetic Code Consortium test cases and canonical solutions
- radar cross section of low observable platforms
- antenna performance
- Microwave monolithic integrated circuit (MMIC) modeling and photonic band gap periodic structures
- electromagnetic environmental effects (E^3), such as EMP, EMI, and compatibility problems
- bioelectromagnetics such as microwave hyperthermia cancer treatment of humans and effects of cellular phones
- multidisciplinary problems invloving optimization of performance for electromagnetics (stealth), aerodynamics (flow management), structures (fatigue and failure), and controls

2. MAXWELL'S EQUATIONS

A general theoretical framework for electromagnetic scattering in the time domain is provided by Maxwell's equations relating the spatial derivatives of the electric and magnetic fields to their time derivatives and to both external and internal sources. In conventional SI notation, the Maxwell curl equations in the presence of polarizable materials are written as:

$$\nabla \times \vec{E} + \frac{\partial \vec{B}}{\partial t} = 0 \quad , \tag{1}$$

$$\nabla \times \vec{H} - \frac{\partial \vec{D}}{\partial t} = \vec{J} \quad , \tag{2}$$

where $\vec{E}$ and $\vec{H}$ are the electric and magnetic field intensities, respectively, while $\vec{D}$ and $\vec{B}$ are the electric displacement and magnetic induction, and $\vec{J}$ is the electric current of "free" charges (to be defined below). These equations are supplemented by two divergence conditions:

$$\nabla \cdot \vec{B} = 0 \quad , \text{ and} \tag{3}$$

$$\nabla \cdot \vec{D} = \rho \quad , \tag{4}$$

where ρ is the volume density of "free" electric charges. Taking the divergence of Eq. (1), one finds that $\partial(\nabla \cdot \vec{B})/\partial t = 0$, so that Eq. (3) can be treated as an initial condition on the fields that will automatically hold at all later times.

Similarly, the divergence of Eq. (2) gives:

$$\nabla \cdot \vec{J} + \partial(\nabla \cdot \vec{D})/\partial t = 0 \quad , \tag{5}$$

while conservation of the "free" electric charge requires that

$$\nabla \cdot \vec{J} + \partial\rho/\partial t = 0 \quad . \tag{6}$$

Combining Eq. (5) and Eq. (6) gives:

$$\partial(\nabla \cdot \vec{D} - \rho)/\partial t = 0 \quad , \tag{7}$$

so that if $\nabla \cdot \vec{D} = \rho$ at some initial instant, Eq. (4) will also hold at all future time.

Each polarizable material is characterized by a volume density of electric dipole moment $\vec{P}$ and magnetic dipole moment $\vec{M}$ in terms of which the electric displacement $\vec{D}$ and the magnetic field intensity $\vec{H}$ can be written as:

$$\vec{D} = \epsilon_0 \vec{E} + \vec{P} \quad , \tag{8}$$

$$\vec{H} = \frac{1}{\mu_0}\vec{B} - \vec{M} \quad , \tag{9}$$

where ϵ_0 is the permittivity of free space ($\simeq 8.854 \times 10^{-12}$ farad/meter in SI units), and μ_0 is the permeability of free space ($= 4\pi \times 10^{-7}$ henry/meter in SI units). The polarization $\vec{P}$ and $\vec{M}$ in general can arise from many types of forces on the material medium, but in radar scattering the most important of these is a linear response to the local electric field $\vec{E}$ and magnetic induction $\vec{B}$. The same is true of the current of "free" charges $\vec{J}$ induced by these fields.

In this limit, the most general form for the response $\vec{R}$ (standing for either $\vec{P}$, $\vec{M}$, or $\vec{J}$) can be written as:

$$\vec{R}(\vec{r},t) = \vec{R}(\vec{r},0) + \int_0^t \left[r_e(\vec{r},t-\tau)\vec{E}(\vec{r},\tau) + r_h(\vec{r},t-\tau)\vec{B}(\vec{r},\tau) \right] d\tau \quad . \tag{10}$$

Here, the properties of the material are lumped into the susceptibilities r_e and r_h. Each r_α is a tensor, reflecting the fact that the response $\vec{R}$ need not be parallel to the applied field. Note also that each r_α depends only on the time difference, $t - \tau$, so that the integral in equation (10) is a simple convolution. The term $\vec{R}(\vec{r},0)$ includes any static, field–independent polarization or current, as one finds in permanent magnets or superconductors, as well as any polarizations due to fields present before $t = 0$.

These additional relations, together with the definitions $\vec{D} = \epsilon_0\vec{E} + \vec{P}$ and $\vec{H} = \frac{1}{\mu_0}\vec{B} - \vec{M}$, constitute a complete linear system of equations that can be solved for the development of $\vec{E}$ and $\vec{B}$ from some initial state that satisfies the subsidiary conditions $\nabla \cdot \vec{B} = 0$ and $\nabla \cdot \vec{D} = \rho_f$. Equations (8), (9), and (10) together already contain implicitly the boundary conditions on the fields that must be satisfied, for instance, at the interface between material ($\vec{P}$ and $\vec{M}$ nonzero) and vacuum ($\vec{P} = \vec{M} = 0$).

In many cases of interest, the time scale for the development of $\vec{E}$ and $\vec{B}$ is much longer than the time it takes for the material to build up its response. In this limit, the relations (10) become approximately instantaneous, and one can write:

$$\begin{aligned} \vec{P} &= p_e\vec{E} + p_m\vec{B} \quad , \\ \vec{M} &= m_e\vec{E} + m_m\vec{B} \quad , \\ \vec{J} &= j_e\vec{E} + j_m\vec{B} \quad , \end{aligned} \tag{11}$$

which greatly simplifies the solution process. For ordinary materials, a further simplification comes from the fact that a magnetic field does not produce either an electric polarization ($p_m = 0$) or a local current ($j_m = 0$), nor does an electric field induce any magnetization ($m_e = 0$). (Materials for which p_m and m_e are nonzero are called <u>chiral</u>.) This allows one to write directly simple expressions for $\vec{D}$, $\vec{H}$, and $\vec{J}$:

$$\begin{aligned} \vec{D} &= \epsilon_0\vec{E} + p_e\vec{E} \triangleq \epsilon\vec{E} \quad , \\ \vec{H} &= \frac{1}{\mu_0}\vec{B} - m_m\vec{B} \triangleq \mu^{-1}\vec{B} \quad , \\ \vec{J} &= \sigma\vec{E} \quad , \end{aligned} \tag{12}$$

in terms of new tensor material parameters ϵ (the permittivity), μ (the permeability), and σ (the conductivity), eliminating the need to calculate $\vec{P}$ and $\vec{M}$ separately. Equation (2) then becomes:

$$\nabla \times \left(\mu^{-1}\vec{B}\right) = \frac{\partial}{\partial t}\left(\epsilon\vec{E}\right) + \sigma\vec{E} \quad , \tag{13}$$

while equation (1) is unchanged.

Because equations (1), (2), and (10) are linear, one can Fourier transform these equations with respect to time, making use of the convolution theorem to obtain a form similar to (11), (12), and (13):

$$\begin{aligned} \widetilde{P}(\omega) &= \tilde{p}_e(\omega)\widetilde{E}(\omega) \quad ; \quad \widetilde{D}(\omega) = \tilde{\epsilon}(\omega)\widetilde{E}(\omega) \\ \widetilde{M}(\omega) &= \tilde{m}_m(\omega)\widetilde{B}(\omega) \quad ; \quad \widetilde{H}(\omega) = \tilde{\mu}^{-1}(\omega)\widetilde{B}(\omega) \\ \nabla \times \widetilde{E} &= i\omega\widetilde{B} \quad ; \quad \nabla \times \widetilde{H} = -i\omega\tilde{\epsilon}\widetilde{E} + \tilde{\sigma}\widetilde{E} \quad , \end{aligned} \tag{14}$$

where we have written $\widetilde{A}(\omega) = \int_{-\infty}^{\infty} e^{i\omega t}A(t)dt$ for every quantity in (1), (2), and (10). This is the standard form in which the material parameters ϵ, μ, and σ are measured and reported, using applied fields of a well defined frequency ω.

3. GENERAL CONSERVATION–LAW FORM

For many problems in mathematical physics, the physical process to be modeled is governed by an appropriate set of linear or nonlinear partial differential equations. For example, many fluid dynamic processes are governed by the Navier–Stokes equations, and the electromagnetic scattering from objects is modeled by Maxwell's equations.

In general, many of these equations naturally lend themselves to a conservation form representation given by:

$$\mathcal{Q}_t + \mathcal{E}_x + \mathcal{F}_y + \mathcal{G}_z = \mathcal{S}(\text{Source}) \quad , \tag{15}$$

where the dependent variable vector $\mathcal{Q}$, and the fluxes $\mathcal{E}$, $\mathcal{F}$, and $\mathcal{G}$ and the source $\mathcal{S}$ take on different forms depending on the physical process being modeled. The integral form of the conservation laws can easily be derived from the differential form by integrating Eq. (15) with respect to x, y, z over any conservation cell whose volume is V:

$$\int\int\int_V \left(\frac{\partial \mathcal{Q}}{\partial t} + \frac{\partial \mathcal{E}}{\partial x} + \frac{\partial \mathcal{F}}{\partial y} + \frac{\partial \mathcal{G}}{\partial z}\right) dx\, dy\, dz = \int\int\int_V \mathcal{S}\, dx\, dy\, dz = \widetilde{\mathcal{S}} \quad . \tag{16}$$

This can be rewritten in vector notation as:

$$\frac{\partial}{\partial t}\int\int\int_V \mathcal{Q}\, dx\, dy\, dz + \int\int\int_V \left(\vec{\nabla}\cdot\vec{F}\right) dx\, dy\, dz = \widetilde{\mathcal{S}} \quad . \tag{17}$$

In the above:

$$\vec{F} = \mathcal{E}\hat{j} + \mathcal{F}\hat{k} + \mathcal{G}\hat{l} \quad . \tag{18}$$

where $\hat{j}$, $\hat{k}$, and $\hat{l}$ are unit vectors along the x, y, and z directions, respectively.

Applying the Gauss divergence theorem, we can convert the volume integral into an integral over the surface A that bounds the volume:

$$\frac{\partial}{\partial t}\left(\widetilde{\mathcal{Q}}V\right) + \int\int_A \left(\vec{F}\cdot\hat{n}\right) dA = \widetilde{\mathcal{S}} \quad . \tag{19}$$

In the above equation, the cell average of the dependent variables is denoted by $\widetilde{\mathcal{Q}}$. The outward unit normal at any point of the boundary surface of a cell has been denoted by $\hat{n} = \hat{n}_x\hat{j} + \hat{n}_y\hat{k} + \hat{n}_z\hat{l}$:

$$\widetilde{\mathcal{Q}} = \frac{\int\int\int_V \mathcal{Q} dV}{\int\int\int_V dV} \quad . \tag{20}$$

The integral form of the conservation laws given by Eq. (19) defines a system of equations for the cell average values of the dependent variables. In order to construct numerical methods to solve the integral form of the conservation laws, one must be able to define cell geometries, approximate the dependent variables, develop spatial discretization procedures, and develop time integration procedures to update the cell averages, etc. There are many

numerical algorithmic issues that come into play in devising a solution procedure to solve either the differential form, Eq. (16), or the integral form, Eq. (19). Some of them are 1) implicit and explicit schemes, 2) stability and order of accuracy, 3) relaxation and approximate factorization procedures, 4) central differenced and upwind schemes, 5) finite–volume and finite difference schemes applied to a structured grid (usually for the differential form), and 6) finite–element–like finite–volume schemes for an unstructured grid (applied to the integral form) setup.

Application of Eq. (15) to many realistic problems requires a coordinate transformation to properly represent the physical domain of interest and to aid in the boundary condition treatment.

Under the transformation of coordinates implied by:

$$\tau = t\,,\ \xi = \xi(t,x,y,z)\,,\ \eta = \eta(t,x,y,z)\,,\ \zeta = \zeta(t,x,y,z)\ ,$$

Eq. (15) can be recast in the conservation form given by:

$$\overline{\mathcal{Q}}_\tau + \overline{\mathcal{E}}_\xi + \overline{\mathcal{F}}_\eta + \overline{\mathcal{G}}_\zeta = \overline{\mathcal{S}} \quad , \tag{21}$$

where:

$$\begin{aligned}
\overline{\mathcal{Q}} &= \frac{\mathcal{Q}}{\mathcal{J}} \quad , \\
\overline{\mathcal{E}} &= \frac{\xi_t}{\mathcal{J}}\mathcal{Q} + \frac{\xi_x}{\mathcal{J}}\mathcal{E} + \frac{\xi_y}{\mathcal{J}}\mathcal{F} + \frac{\xi_z}{\mathcal{J}}\mathcal{G} \quad , \\
\overline{\mathcal{F}} &= \frac{\eta_t}{\mathcal{J}}\mathcal{Q} + \frac{\eta_x}{\mathcal{J}}\mathcal{E} + \frac{\eta_y}{\mathcal{J}}\mathcal{F} + \frac{\eta_z}{\mathcal{J}}\mathcal{G} \quad , \\
\overline{\mathcal{G}} &= \frac{\zeta_t}{\mathcal{J}}\mathcal{Q} + \frac{\zeta_x}{\mathcal{J}}\mathcal{E} + \frac{\zeta_y}{\mathcal{J}}\mathcal{F} + \frac{\zeta_z}{\mathcal{J}}\mathcal{G} \quad , \\
\overline{\mathcal{S}} &= \frac{\mathcal{S}}{\mathcal{J}} \quad ,
\end{aligned} \tag{22}$$

where, in turn, $\mathcal{J}$ is the Jacobian of the transformation:

$$J = \partial(\tau,\xi,\eta,\zeta)/\partial(t,x,y,z) = \frac{1}{V} \tag{23}$$

and ξ_t, ξ_x, ξ_y, η_t, η_x, η_y, η_z, ζ_t, ζ_x, ζ_y, and ζ_z are the transformation metrics.

Associating the subscripts j,k,l with the ξ,η,ζ directions, a numerical approximation to Eq. (21) (as well as Eq. (19)) may be expressed in the semidiscrete conservation law form given by:

$$\begin{aligned}
\left(\left(\widetilde{\mathcal{Q}}V\right)_{j,k,l}\right)_\tau &+ \left(\widehat{\mathcal{E}}_{j+1/2,k,l} - \widehat{\mathcal{E}}_{j-1/2,k,l}\right) \\
&+ \left(\widehat{\mathcal{F}}_{j,k+1/2,l} - \widehat{\mathcal{F}}_{j,k-1/2,l}\right) \\
&+ \left(\widehat{\mathcal{G}}_{j,k,l+1/2} - \widehat{\mathcal{G}}_{j,k,l-1/2}\right) = \widetilde{\mathcal{S}} \quad ,
\end{aligned} \tag{24}$$

where $\widehat{\mathcal{E}}, \widehat{\mathcal{F}}, and \widehat{\mathcal{G}}$ are the numerical fluxes representing the physical fluxes $\overline{\mathcal{E}}$, $\overline{\mathcal{F}}$, and $\overline{\mathcal{G}}$ at the bounding sides of the cell for which discrete conservation is considered, and $\widetilde{\mathcal{Q}}_{j,k,l}$ is the cell average of the dependent variables. The half–integer subscripts denote cell sides and the integer subscripts the cell itself or its centroid.

The semidiscrete conservation law form given by Eq. (24) may be regarded as representing a finite volume discretization because the Jacobian $\mathcal{J}$ appearing in Eq. (23) is associated with the volume of the cell, and the metrics $\frac{\xi_x}{\mathcal{J}}$, $\frac{\xi_y}{\mathcal{J}}$, and so on appearing in the flux terms (Eq. (22)) are nothing but the components of the appropriate cell surface area.

The objective is to solve Eq. (24) for the dependent vector $\widetilde{\mathcal{Q}}$. After incorporation of proper flux representation, the discrete form of Eq. (24) can be written as:

$$R(\mathcal{Q}) = 0 \quad . \tag{25}$$

If $\mathcal{Q}$ is sought in the neighborhood of a known solution $\mathcal{Q}^*$, then solution to Eq. (25) can be written as:

$$\frac{\partial R}{\partial \mathcal{Q}}(\mathcal{Q} - \mathcal{Q}^*) = -R(\mathcal{Q}^*) \quad , \tag{25a}$$

where $\frac{\partial R}{\partial \mathcal{Q}}$, in general, is a differential operator. All of the above–mentioned algorithmic issues apply to how one models the $\frac{\partial R}{\partial \mathcal{Q}}$ operator. References 1–29 provide many of the algorithmic details.

Unstructured Grid Approach

In the unstructured grid approach, there is no defined set of coordinates, such as the ξ, η, ζ system in the differential approach. The cell shape is arbitrary and can be made up of any polyhedron shape such as a hexahedron, prism, or a tetrahedron. The coordinate directions for each cell are defined by its cell interface normals, and one will employ the integral conservation form of equations, Eq. (19), to enforce the flux conservation.

We first divide the surface integral into component parts that apply over each distinct face or side of any cell under consideration. We then replace the integral on each face with a numerical quadrature as part of the numerical approximation procedure.

$$\begin{aligned} \int\int_S (\vec{F}\cdot\hat{n})\,dS &= \sum_{\text{faces}} \int\int_F (\vec{F}\cdot\hat{n})\,dS \\ &= \sum_{\text{faces}} \sum_{\text{quads.}} \vec{F}_i \cdot \hat{n}_i\, S_i \end{aligned} \tag{26a}$$

Here, "quads." is an abbreviation for "quadrature points." The weights of the quadrature formulae must include the effect of cell surface area corresponding to the given face and such weights are denoted by S_i in the above equation. The midpoint formula can be represented as

$$\int\int_S (\vec{F}\cdot\hat{n})\,dS = \sum_{\text{faces}} \vec{F}_m \cdot \hat{n}_m\, S_m \tag{26b}$$

where m denotes the centroid of each face. Even higher-order quadrature formulae may be used if necessary.

The original initial value problem (IVP) for the differential form of the conservation laws specifies initial values of the dependent variables. In the IVP for the integral form of the conservation laws, initial values of cell averages of the dependent variables will be defined. Given such initial values, the three steps used to set up the discretization procedure for the integral form of the conservation laws are as follows.

(i) Define dependent variable polynomials in each cell so that the cell average of the polynomial approximation matches the cell average of the dependent variable which is either given as part of the initial value specification or obtained by updating the cell averages during subsequent steps of the solution process. This process of defining pointwise polynomial behavior from known values of cell averages is called the "reconstruction procedure". At quadrature points on boundary faces and at other locations, it may also be required to construct the polynomials by fitting them to known pointwise values in addition to cell average values of the dependent variables.

(ii) Evaluate the polynomials at all quadrature points. This will lead to "left" and "right" values at each quadrature point which lies on a face common to two cells.

(iii) Construct the solution of a local Riemann problem using these left and right states and from this evaluate the numerical flux. Use such numerical fluxes in Eq. 26a or Eq. 26b.

(iv) Steps (i)–(iii) complete the discretization of the right hand side of Eq. 19. To the resulting semi-discrete system of equations, we apply a suitable time-integration procedure.

Features of Structured Grid–Based CEM

- Maxwell's equations in differential conservation form
- Explicit Lax–Wendroff upwind finite–volume scheme
- Multizone structured gridding
- Convenient bookkeeping/data structure
- Efficiently vectorizable

Disadvantages

- Multizone gridding process is tedious and time consuming
- Zonal interface boundary conditions degrade execution efficiency
- Difficult to grid certain interior regions

Features of Unstructured Grid–Based CEM

- Maxwell's equations in integral conservation form
- Any region can be gridded
- Higher–order basis functions for solution variables in each cell
- Well suited for MIMD architectures
- Can handle special regions (thin wire, crack, $\cdots$) through choice of basis functions

Disadvantages

- Accuracy may be an issue for arbitrarily arranged unstructured cells
- Vectorization may be difficult

Unstructured Grid–Based CEM Issues

- Geometry/unstructured grid generation
  - — surface definition
  - — surface grid
  - — volume grid
- Integral equation based finite–volume scheme
  - — basis function in each cell
  - — flux evaluation at interface
  - — time discretization
- Graphics/visualization

In the development of an unstructured grid–based CEM method, the present work addresses some of these issues.

4.0 CONSERVATION FORM FOR MAXWELL'S EQUATIONS

We can regard equations (1) and (2) as a first–order system of partial differential equations for the time development of the dependent variables $\vec{B}$ and $\vec{D}$:

$$\frac{\partial}{\partial t}\begin{pmatrix}\vec{B}\\ \vec{D}\end{pmatrix}+\begin{pmatrix}\nabla\times\vec{E}\\ -\nabla\times\vec{H}\end{pmatrix}=\begin{pmatrix}0\\ -\vec{J}\end{pmatrix} \quad . \tag{27}$$

At any instant the functions $\vec{E}$, $\vec{H}$, and $\vec{J}$ can be derived from the current values of $\vec{D}$ and $\vec{B}$, and by substitution into Eq. (27) they determine new values for the time derivatives.

In order to apply CFD–based conservation–law form finite–volume methods, Eq. (27) is rewritten in the form of Eq. (15), where:

$$\mathcal{Q}=\begin{Bmatrix}B_x\\ B_y\\ B_z\\ D_x\\ D_y\\ D_z\end{Bmatrix};\ \mathcal{E}=\begin{Bmatrix}0\\ -D_z/\epsilon\\ D_y/\epsilon\\ 0\\ B_z/\mu\\ -B_y/\mu\end{Bmatrix};\ \mathcal{F}=\begin{Bmatrix}D_z/\epsilon\\ 0\\ -D_x/\epsilon\\ -B_z/\mu\\ 0\\ B_x/\mu\end{Bmatrix};\ \mathcal{G}=\begin{Bmatrix}-D_y/\epsilon\\ D_x/\epsilon\\ 0\\ B_y/\mu\\ -B_x/\mu\\ 0\end{Bmatrix};\ \mathcal{S}=\begin{Bmatrix}0\\ 0\\ 0\\ -J_x\\ -J_y\\ -J_z\end{Bmatrix} \quad . \tag{28}$$

If we define unit vectors $\hat{x},\hat{y},\hat{z}$ along three orthogonal coordinate directions in space, equation (28) can be written in a compact form as:

$$\frac{\partial}{\partial t}\begin{pmatrix}\vec{B}\\ \vec{D}\end{pmatrix}+\frac{\partial}{\partial x}\begin{pmatrix}\hat{x}\times\vec{E}\\ -\hat{x}\times\vec{H}\end{pmatrix}+\frac{\partial}{\partial y}\begin{pmatrix}\hat{y}\times\vec{E}\\ -\hat{y}\times\vec{H}\end{pmatrix}+\frac{\partial}{\partial z}\begin{pmatrix}\hat{z}\times\vec{E}\\ -\hat{z}\times\vec{H}\end{pmatrix}=\begin{pmatrix}0\\ -\vec{J}\end{pmatrix} \quad , \tag{29}$$

in which every one of the six component equations is analogous to the conservation law $\partial\rho/\partial t+\nabla\cdot J=0$ which holds for electric charge. In a formal sense, one can consider the set of six unknowns $(\vec{B},\vec{D})$ as a "vector" density $\mathcal{Q}$, and define a "tensor" flux F with components:

$$F_x=\begin{pmatrix}\hat{x}\times\vec{E}\\ -\hat{x}\times\vec{H}\end{pmatrix} \quad , \quad F_y=\begin{pmatrix}\hat{y}\times\vec{E}\\ -\hat{y}\times\vec{H}\end{pmatrix} \quad , \quad F_z=\begin{pmatrix}\hat{z}\times\vec{E}\\ -\hat{z}\times\vec{H}\end{pmatrix} \tag{30}$$

such that equation (29) becomes:

$$\partial Q/\partial t+\nabla\cdot F=\begin{pmatrix}0\\ -\vec{J}\end{pmatrix}\triangleq\mathcal{S} \quad . \tag{31}$$

This rather abstract representation as a "system of conservation laws" provides a convenient starting point for the development of integration algorithms.

Also, one can transform the equations to arbitrary curvilinear coordinates (ξ,η,ζ) without alteration to this basic form:

$$\frac{\partial}{\partial t}\left(\frac{\mathcal{Q}}{\mathcal{J}}\right)+\frac{\partial}{\partial\xi}\begin{pmatrix}\vec{\xi}\times\vec{E}/\mathcal{J}\\ -\vec{\xi}\times\vec{H}/\mathcal{J}\end{pmatrix}+\frac{\partial}{\partial\eta}\begin{pmatrix}\vec{\eta}\times\vec{E}/\mathcal{J}\\ -\vec{\eta}\times\vec{H}/\mathcal{J}\end{pmatrix}+\frac{\partial}{\partial\zeta}\begin{pmatrix}\vec{\zeta}\times\vec{E}/\mathcal{J}\\ -\vec{\zeta}\times\vec{H}/\mathcal{J}\end{pmatrix}=\frac{\mathcal{S}}{\mathcal{J}} \quad , \tag{32}$$

where $\mathcal{J}$ is the Jacobian of the coordinate transformation and, for example, $\vec{\xi} = (\partial\xi/\partial x, \partial\xi/\partial y, \partial\xi/\partial z)$.

In addition to the differential form, one can use the divergence theorem on any volume V to obtain the integral form of Maxwelll's equations:

$$\frac{\partial}{\partial t}\int\int_V\int \begin{pmatrix}\vec{B}\\ \vec{D}\end{pmatrix} dV + \int_A\int \begin{pmatrix}\hat{n}\times\vec{E}\\ -\hat{n}\times\vec{H}\end{pmatrix} dA = \int\int_V\int \mathcal{S}\, dV \quad , \tag{33}$$

where the six components of $\vec{F}\cdot\hat{n}$ in Eq. (19) are $(\hat{n}\times E, -\hat{n}\times H)$.

Equation (33) replaces all the spatial derivatives with area integrals over the surface of V.

4.1 Interface Flux Forms

In order to evaluate the fluxes $\widehat{\mathcal{E}}$, $\widehat{\mathcal{F}}$, and $\widehat{\mathcal{G}}$ required by a finite–volume procedure, Eq. (24), at appropriate cell boundaries, first the characteristics of Eq. (32) are analyzed in each (τ,ξ), (τ,η), and (τ,ζ) plane. For example, the characteristics (eigenvalues) in the (τ,ξ) plane are given by solving the matrix equation $|A-\lambda I| = 0$, where $A = \partial\overline{\mathcal{E}}/\partial\overline{\mathcal{Q}}$ is the Jacobian matrix.

For a locally isotropic medium, $\vec{D} = \epsilon\vec{E}$ and $\vec{B} = \mu\vec{H}$, the matrix A has six eigenvalues: two vanishing ($\lambda_{1,2} = 0$), two equal to $c|\vec{\xi}\,|$, and two equal to $-c|\vec{\xi}\,|$, where $c = 1/\sqrt{\mu\epsilon}$ and $|\vec{\xi}\,| = \sqrt{\xi_x^2+\xi_y^2+\xi_z^2}$. Referring to Fig. 4.1.1, the negative and positive eigenvalues are indicated at a cell boundary (interface). The intent is to compute the interface flux given the left state $-$ and the right state $+$ on either side of the interface. For a linear system, across a characteristic λ, the variation in $\overline{\mathcal{Q}}$ and $\overline{\mathcal{E}}$ are related by a jump condition $-\lambda|\overline{\mathcal{Q}}\,|+|\overline{\mathcal{E}}\,| = 0$, where $|\quad|$ denotes jump. To make the treatment more general, a resistive sheet is introduced at a cell boundary (Fig. 4.1.1) whose resistivity is denoted by $(1/\sigma d)$ where σ is the conductivity and d is the thickness of the sheet. The resistive sheet can allow for total tangential magnetic fields $\left(\vec{\xi}\times H\right)$ to jump across the sheet. Corresponding to the ξ–direction interface, the following jump relationships are written[14].

Across $-$ Characteristics $\left(\lambda = -c|\vec{\xi}\,|\,,\ \widehat{\xi} = \vec{\xi}/|\vec{\xi}\,|\right)$

$$(B^* - B^-) = -\frac{\widehat{\xi}}{c^-}\times(E^* - E^-) \tag{34a}$$

Across $+$ Characteristics

$$(B^+ - B^{**}) = \frac{\widehat{\xi}}{c^+}\times(E^+ - E^{**}) \tag{34b}$$

At the Interface ($\lambda = 0$) (Allowing for a Resistive Card)

$$\widehat{\xi}\times(E^{**} - E^*) = 0 \tag{34c}$$

$$\widehat{\xi} \times (H^{**} - H^{*}) = -\sigma d \left(\widehat{\xi} \times \widehat{\xi} \times E^{*} \right) \tag{34d}$$

Relationship Eq. (34d) is obtained by integrating Eq. (32) over an infinitesimal strip area enclosed by a contour containing the resistive sheet. Solving the above four relationships, one can write the expressions for the interface fluxes:

$$\begin{aligned}
\widehat{\xi} \times E^{*} &= \widehat{\xi} \times \frac{\left\{ \left[E^{+}(\epsilon c)^{+} + \widehat{\xi} \times H^{+} \right] + \left[(\epsilon c)^{-} E^{-} - \widehat{\xi} \times H^{-} \right] \right\}}{(\epsilon c)^{-} + (\epsilon c)^{+} + \sigma d} \\
\widehat{\xi} \times H^{*} &= \widehat{\xi} \times \frac{\left\{ \left[1 + \sigma d (\mu c)^{+} \right] \left(H^{-} (\mu c)^{-} + \widehat{\xi} \times E^{-} \right) + \left[(\mu c)^{+} H^{+} - \widehat{\xi} \times E^{+} \right] \right\}}{(\mu c)^{+} + (\mu c)^{-} + \sigma d (\mu c)^{+} (\mu c)^{-}} \\
\widehat{\xi} \times H^{**} &= \widehat{\xi} \times \frac{\left\{ \left[1 + \sigma d (\mu c)^{-} \right] \left((\mu c)^{+} H^{+} - \widehat{\xi} \times E^{+} \right) + \left[(\mu c)^{-} H^{-} + \widehat{\xi} \times E^{-} \right] \right\}}{(\mu c)^{+} + (\mu c)^{-} + \sigma d (\mu c)^{-} (\mu c)^{+}}
\end{aligned} \tag{35}$$

The superscripts + and − denote the right and left state at a cell interface. When $\sigma d = 0$, the total tangential electric and magnetic fields are continuous across a material interface. When $\sigma d \rightarrow \infty$, the resistive sheet represents a perfectly conducting surface, and the total tangential electric field $\widehat{\xi} \times E^{*}$ goes to zero satisfying Maxwell's boundary conditions for a perfectly conducting surface (to be discussed later). The corresponding tangential magnetic fields $\left(\widehat{\xi} \times H^{*} \right)$ depend only on E^{-} and H^{-}, and similarly $\left(\widehat{\xi} \times H^{**} \right)$ depends only on E^{+} and H^{+}. Equation (35) allows for the material properties (ϵ, μ) to jump any amount at an interface. In free space where c^{+}, c^{-}, μ^{+}, μ^{-}, ϵ^{+}, and ϵ^{-} are normalized to unity, Eq. (35) reduces to a simpler form.

Given a left (−) state and a right (+) state, the process of computing the intermediate states (* and **) is usually referred to as the "Riemann" problem.

Additional information on these interface flux forms, and on the eigenvalue/eigenvector structure of the Maxwell equations is given in Section 4.

4.2 Characteristics of Maxwell's Equations

The characteristics of Eq. (21) are analyzed by considering the Jacobians $\frac{\partial \mathcal{E}}{\partial Q}$, $\frac{\partial \mathcal{F}}{\partial Q}$, and $\frac{\partial \mathcal{G}}{\partial Q}$. For example, the Jacobian matrix $\frac{\partial \mathcal{E}}{\partial Q}$ for Maxwell's equations, Eq. (32), becomes:

$$A = \frac{\partial \mathcal{E}}{\partial Q} = \begin{bmatrix} 0 & 0 & 0 & 0 & -\xi_z/\mu & \xi_y/\mu \\ 0 & 0 & 0 & \xi_z/\mu & 0 & \xi_x/\mu \\ 0 & 0 & 0 & -\xi_y/\mu & \xi_x/\mu & 0 \\ 0 & \xi_z/\epsilon & -\xi_y/\epsilon & 0 & 0 & 0 \\ -\xi_z/\epsilon & 0 & \xi_x/\epsilon & 0 & 0 & 0 \\ \xi_y/\epsilon & -\xi_x/\epsilon & 0 & 0 & 0 & 0 \end{bmatrix} \tag{36}$$

which can be symbolically written as:

$$\frac{\partial \mathcal{E}}{\partial Q} = \begin{pmatrix} 0 & -(\vec{\xi}/\mu)\times \\ (\vec{\xi}/\epsilon)\times & 0 \end{pmatrix} \quad , \tag{37}$$

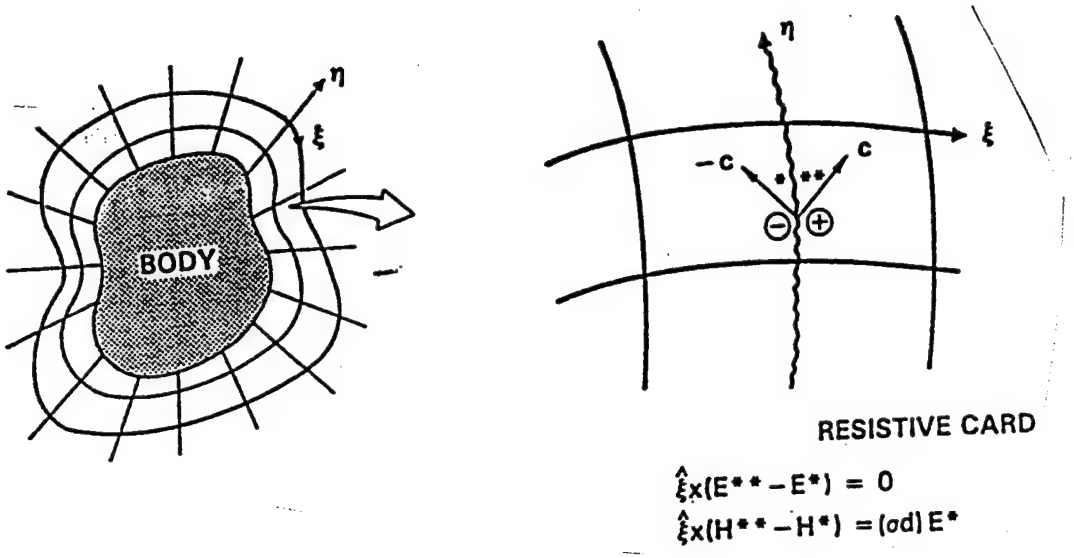


Fig. 4.1.1. Interface Flux Treatment with a Resistive Sheet.

where:

$$\left(\frac{\vec{\xi}}{\mu}\right) \times = \begin{bmatrix} 0 & \xi_z/\mu & -\xi_y/\mu \\ -\xi_z/\mu & 0 & \xi_x/\mu \\ \xi_y/\mu & -\xi_x/\mu & 0 \end{bmatrix} \quad . \tag{38}$$

The eigenvalues of the Jacobian matrix A in Eq. (36) are obtained from solving:

$$|A - \lambda\, I| = 0 \quad . \tag{39}$$

and the right eigenvector corresponding to a kth eigenvalue, λ_k, is obtained by solving:

$$|A - \lambda_k I|\{r_k\} = 0 \quad . \tag{40}$$

For Maxwell's equations. solution to Eq. (39) reuslts in six eigenvalues: two vanishing, two equal to $c|\vec{\xi}|$, and two equal to $-c|\vec{\xi}|$, where $c = 1/\sqrt{\mu\epsilon}$, $\vec{\xi} = (\xi_x, \xi_y, \xi_z)$, and $|\vec{\xi}| = \sqrt{\xi_x^2 + \xi_y^2 + \xi_z^2}$.

Defining the upper and lower halves of the eigenvector $\vec{r}$ as $\vec{e}$ and $\vec{h}$, respectively, so that $\vec{r} = (\vec{e}, \vec{h})^T$, one finds corresponding to the zero eigenvalues:

$$v_1 = \begin{pmatrix} 0 \\ \widehat{\xi} \end{pmatrix} \quad , \quad v_2 = \begin{pmatrix} \widehat{\xi} \\ 0 \end{pmatrix} \quad , \tag{41}$$

where $\widehat{\xi} = \vec{\xi}/|\vec{\xi}|$. Corresponding to the positive or negative eigenvalues, one finds that $\vec{e}$, $\vec{h}$, and $\vec{\xi}$ form a mutually orthogonal system with $\vec{h} = \pm(\widehat{\xi} \times \vec{e}/\epsilon c)$, and where the upper sign goes with the negative eigenvalue. For example, corresponding to the two negative eigenvalues one can construct the two independent set of eigenvectors by choosing

$\vec{e} = (x_\eta, y_\eta, z_\eta)$, a tangent vector in the direction η on a constant–ξ plane for one, and $\vec{e} = (x_\zeta, y_\zeta, z_\zeta)$, a tangent vector in the direction ζ for the other. Once $\vec{e}$ is chosen $\vec{h}$ can be obtained using the mutually orthogonal property $\vec{h} = (\hat{\xi} \times \vec{e}/\epsilon c)$.

For hyperbolic systems of equations, the Jacobian has real eigenvalues and a linearly independent set of eigenvectors[14,31]. The equations are also characterized as "linear" if:

$$\nabla_Q \lambda_k \cdot r_k = 0 \tag{42}$$

and "nonlinear" if:

$$\nabla_Q \lambda_k \cdot r_k \neq 0 \quad . \tag{43}$$

It can be verified that time–domain Maxwell's equations are hyperbolic and linear. For a linear, hyperbolic equation of the form

$$q_t + f_x = 0 \tag{44}$$

one can write the jump condition[14]:

$$-\lambda[q] + [f] = 0 \tag{45}$$

where [q] and [f] are the changes across an eigenvalue λ. This jump condition is used in deriving the interface flux forms in Section 3.2.

For a nonlinear system, the jump condition is defined only across a surface of discontinuity $S(x,t) = 0$, such that:

$$S_t[q] + S_x[f] = 0 \tag{46}$$

where $(S_t/S_x) = -(dx/dt) = \Lambda$ is the characteristic speed of the discontinuity. For a linear system Λ is replaced by λ, the eigenvalue.

4.3 The Riemann Solver: Preliminaries

The wave solutions of Maxwell's equations can be derived in a different way by looking for characteristic combinations of $\vec{E}$ with $\vec{H}$ (or $\vec{D}$ with $\vec{B}$) that propagate without change along a particular direction in space. The form of these combinations can be illustrated using one space dimension (say, x), in which case Maxwell's equations reduce to:

$$\frac{\partial B_z}{\partial t} + \frac{\partial E_y}{\partial x} = 0 \quad , \quad \frac{\partial D_y}{\partial t} + \frac{\partial H_z}{\partial x} = 0 \quad . \tag{47}$$

In free space, $E_y = D_y/\epsilon_0$ and $H_z = B_z/\mu_0$, so the two equations can be rewritten in matrix form as:

$$\frac{\partial}{\partial t}\begin{pmatrix} B_z \\ D_y \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} 0, \epsilon_0^{-1} \\ \mu_0^{-1}, 0 \end{pmatrix}\begin{pmatrix} B_z \\ D_y \end{pmatrix} = 0 \quad , \tag{48a}$$

or:

$$\frac{\partial}{\partial t}Q + \frac{\partial}{\partial x}AQ = 0, \text{ where } Q = (B_z, D_y) \quad . \tag{48b}$$

By direct substitution into (47), it is easy to show that:

$$\frac{\partial}{\partial t}\left(B_z - \sqrt{\frac{\mu_0}{\epsilon_0}}D_y\right) = c\frac{\partial}{\partial x}\left(B_z - \sqrt{\frac{\mu_0}{\epsilon_0}}D_y\right) \quad , \tag{49a}$$

$$\frac{\partial}{\partial t}\left(B_z + \sqrt{\frac{\mu_0}{\epsilon_0}}D_y\right) = -c\frac{\partial}{\partial x}\left(B_z + \sqrt{\frac{\mu_0}{\epsilon_0}}D_y\right) \quad , \tag{49b}$$

where $c = 1/\sqrt{\mu_0\epsilon_0}$ is the velocity of light in vacuum.

These are just two scalar, first–order equations having the general solutions:

$$B_z + \sqrt{\frac{\mu_0}{\epsilon_0}}D_y = g_1(x - ct) \quad , \tag{50a}$$

$$B_z - \sqrt{\frac{\mu_0}{\epsilon_0}}D_y = g_2(x + ct) \quad , \tag{50b}$$

the first representing a wave traveling toward increasing values of x and the second, a wave traveling in the opposite direction. The quantity $\zeta_0 \triangleq \sqrt{\mu_0/\epsilon_0}$ is known as the free–space impedance.

From (50) the most general solution of the one–dimensional Maxwell's equations can be written as:

$$B_z = g_1(x - ct) + g_2(x + ct) \tag{51a}$$

$$D_y = [g_1(x - ct) - g_2(x + ct)]\,/\zeta_0 \quad . \tag{51b}$$

So, whatever initial distribution of B and D we are given at $t = 0$, we can decompose it uniquely into a right–moving and a left–moving wave, each of which travels at the velocity of light. In the (x, t) plane, the right–moving combination $g_1 = B + \zeta_0 D$ is seen to be constant along the characteristic lines $x - ct =$ const., while $g_2 = B - \zeta_0 D$ is constant along the lines $x + ct =$ const. Thus, the value of B at any point in the upper (x, t) plane can be constructed as shown in the figure from the initial values of B and D at $x_0 = x - ct$ and $x_1 = x + ct$.

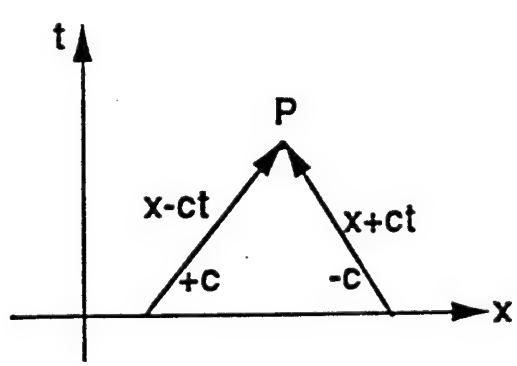


Fig. 4.2.1 Right and left moving characteristics.

Another way to view the characteristic combinations $g_1 = B + \zeta_0 D$ and $g_2 = B - \zeta_0 D$ is as defining eigenvectors of the 2×2 matrix A in equation (48):

$$(B, D) = (1, +\zeta_0^{-1})K \text{ for right moving waves} \tag{52a}$$

$$(B, D) = (1, -\zeta_0^{-1})K \text{ for left moving waves} \tag{52b}$$

$$\begin{pmatrix} 0 & \epsilon_0^{-1} \\ \mu_0^{-1} & 0 \end{pmatrix} \begin{pmatrix} 1 \\ +\zeta_0^{-1} \end{pmatrix} = \begin{pmatrix} 1/\zeta_0\epsilon_0 \\ 1/\mu_0 \end{pmatrix} = +c \begin{pmatrix} 1 \\ -\zeta_0^{-1} \end{pmatrix} \tag{53a}$$

$$\begin{pmatrix} 0 & \epsilon_0^{-1} \\ \mu_0^{-1} & 0 \end{pmatrix} \begin{pmatrix} 1 \\ -\zeta_0^{-1} \end{pmatrix} = \begin{pmatrix} -1/\zeta_0\epsilon_0 \\ 1/\mu_0 \end{pmatrix} = -c \begin{pmatrix} 1 \\ -\zeta_0^{-1} \end{pmatrix} \quad . \tag{53b}$$

This reveals a fundamental connection between the abstract representation as a system of conservation laws and the propagation characteristics of the original problem: the eigenvectors of the abstract system (known as the Riemann invariants) provide a way to decompose the initial data that allows us to construct the solution at all later times.

For a linear hyperbolic equation of the form $q_t + f_x = 0$, one can diagonalize the Jacobian matrix $A = \frac{\partial f}{\partial q}$ such that $A = R[\lambda]L$, where L is the left eigenvector matrix, R is the right eigenvector matrix, and $[\lambda]$ is the diagonal matrix made up of eigenvalues. Multiplying throughout by L, one can write the equation $(Lq)_t + [\lambda](Lq)_x = 0$, where now the combination (Lq) is the Riemann invariant.

4.4 Discretization and Dissipation

The true initial–value problem we must solve numerically is to determine the behavior of the electromagnetic fields on a discrete mesh, from data originally prescribed on this same mesh. In one dimension, the fields $B(x,t)$ and $D(x,t)$ are represented by values $b_i(t)$ and $d_i(t)$ attributed to each interval of the space coordinate x that lies within the computational domain (say, $0 \leq x < L$). In our finite–volume approach, these values can be thought of either as approximating the true values of B and D at the center of each interval at a given time, or as the averages of B and D over the interval. More generally, B may be evaluated on a different spatial mesh from D, or the behavior of B and D within each interval may be modeled by high–order polynomials.

In any case, one cannot directly solve the partial differential equations for B and D, but instead must seek a discrete set of equations for $\{b_i\}$ and $\{d_i\}$ whose solution for any later time approaches that of the original Maxwell's equations as the mesh is refined. This is the basic concern of all numerical integration techniques. In what follows, we describe a particular approach to this problem that uses the Riemann invariants to achieve both accuracy and stability in the integration process.

This method was developed primarily by S. Osher[31] and various co–workers, including Sukumar Chakravarthy of the Rockwell Science Center. They considered the general hyperbolic system:

$$\frac{\partial w}{\partial t} + \frac{\partial}{\partial x} f(w) = 0 \quad , \tag{54}$$

where $w(x,t)$ is a vector of unknown fields and the flux $f(w)$ is a vector–valued function whose Jacobian matrix $\partial f/\partial w$ has only real eigenvalues.

Discretizing the space variable in this equation forces one to replace the spatial derivative with an approximation based on finite differences. The order of this approximation, in the sense of Taylor series expansion, is then found to determine the order of accuracy of the computed fields. Furthermore, the details of this approximation place limits on the stability of any scheme chosen to integrate (54) forward in time. In particular, we do not want the ragged form of the discretized initial data $w_i(0)$ to give rise to increasing raggedness in the solution $w_i(t)$ at later times.

What Osher did, in essence, was to rewrite (54) in terms of its Riemann invariant combinations, and determine for each invariant an appropriate discretized flux. Just as in our 1D Maxwell example, this flux carries information only from the past locations of the wave to the present, and it is therefore called an "upwind" approximation.

The first step in Osher's procedure is to find the eigenvectors r_k of $\partial f/\partial w$, together with their associated eigenvalues λ_k. As we have seen in the example, these eigenvectors are the combinations of the components of w that propagate locally as simple waves $g_k(x-\lambda_k t)$:

$$\left[\frac{\partial f}{\partial w}\right] r_k = \lambda_k r_k \quad . \tag{55}$$

For the special case of a linear dependence of f on w, we note that the matrix $\partial f/\partial w$ is constant, and the eigenvectors and eigenvalues are independent of w.

The second step is to construct a two–point approximation $h_+(w_j, w_{j+1})$ for the flux entering the jth cell from the interface between cells j and $j+1$. Osher does this by finding a value $w^*_{j+1/2}$ that is consistent with data propagated from the interior of both neighboring cells and putting:

$$h_+(w_j, w_{j+1}) = f(w^*_{j+1/2}) \quad . \tag{56}$$

A similar construction at the $(j-1, j)$ interface yields a value $w^{**}_{j-1/2}$ consistent with data in cells $j-1$ and j and an approximation:

$$h_-(w_{j-1}, w_j) = f(w^{**}_{j-1/2}) \tag{57}$$

for the flux entering the jth cell from the $j-1/2$ interface. These approximations can be used directly to obtain a first–order accurate approximation to (54):

$$\frac{dw_j}{dt} + \frac{1}{\Delta}\left[f(w^*_{j+1/2}) - f(w^{**}_{j-1/2})\right] = 0 \quad . \tag{58}$$

More importantly, the first–order fluxes form the basis for constructing integration schemes of higher order, such as the Lax–Wendroff procedure used in this report.

How, then, are these interface values of w constructed from the Riemann invariants? In Osher's method an integration is performed in the abstract space of the possible values of

w, starting (in the first case) from w_j and ending at w_{j+1}, and following a path determined by those eigenvectors that represent waves propagating toward from the interface from either side. The first leg of this integration puts:

$$\frac{dw}{ds_1} = r_1\big(w(s_1)\big) \quad , \tag{59}$$

where r_1 is the eigenvector that corresponds to the wave traveling at the greatest speed $\left(\triangleq \lambda_1 < 0\right)$ away from the interface from the interior of the jth cell. The first leg ends at a point $w_{j,1}$ which we have to leave undetermined until the path finally reaches w_{j+1}. The second leg puts:

$$\frac{dw}{ds_2} = r_2\big(w(s_2)\big) \tag{60}$$

and so forth, until all the waves propagating from $j + 1/2$ to j have been used. The (as yet undetermined) value of w at this end point is $w^*_{j+1/2}$. Now the waves for which $\lambda_k = 0$ are used in the same fashion, *i.e.*, one integrates

$$\frac{dw}{ds_k} = r_k\big(w(s_k)\big) \tag{61}$$

along each such eigenvector. The end point of this series of integrations will be $w^{**}_{j+1/2}$.

The last series of legs involve the waves that propagate toward the interior of cell $j+1$ from the $j + 1/2$ interface, done in the order from smallest speed to largest. Finally, on the last leg one has:

$$\frac{dw}{ds_m} = r_m\big(w(s_m)\big) \quad , \tag{62}$$

and the final end point is w_{j+1}.

The m known components of w_j and w_{j+1} provide just enough information to determine the m scalar quantities $\{s_k\}$ and therefore the interface values $w^*_{j+1/2}$ and $w^{**}_{j+1/2}$.

As an important illustration of this procedure, consider the one–dimensional Maxwell's equations that describe an inhomogeneous medium, that is, a medium in which ϵ and μ are functions of x:

$$\frac{\partial}{\partial t}\begin{pmatrix} B \\ D \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} 0, \epsilon^{-1} \\ \mu^{-1}, 0 \end{pmatrix}\begin{pmatrix} B \\ D \end{pmatrix} = 0 \quad . \tag{63}$$

If we take w to be simply (B, D) then this equation is not in the form of (54) because the flux depends explicitly on x through ϵ and μ. To get around this difficulty, we define ϵ^{-1} and μ^{-1} as additional dependent variables e and m, and write:

$$\frac{\partial}{\partial t}\begin{pmatrix} B \\ D \\ e \\ m \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} eD \\ mB \\ 0 \\ 0 \end{pmatrix} = 0 \quad . \tag{64}$$

Then the Jacobian matrix becomes:

$$\frac{\partial f}{\partial w} = \begin{pmatrix} 0 & e & D & 0 \\ m & 0 & 0 & B \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad , \tag{65}$$

which has two zero eigenvalues $\lambda_2 = \lambda_3 = 0$, one positive eigenvalue $\lambda_4 = c = em$ and one negative, $\lambda_1 = -c$. The corresponding eigenvectors are:

$$r_1 = \begin{pmatrix} c \\ -m \\ 0 \\ 0 \end{pmatrix} \quad ; \quad r_2 = \begin{pmatrix} 0 \\ D \\ -e \\ 0 \end{pmatrix} \quad ; \quad r_3 = \begin{pmatrix} B \\ 0 \\ 0 \\ -m \end{pmatrix} \quad ; \quad r_4 = \begin{pmatrix} e \\ c \\ 0 \\ 0 \end{pmatrix} \quad . \tag{66}$$

Along the first leg of integration, we now have:

$$\frac{dB}{ds} = c \quad ; \quad \frac{dD}{ds} = -m \quad ; \quad \frac{de}{ds} = 0 = \frac{dm}{ds} \tag{67}$$

$$B^*_{j+1/2} - B_j = cs \quad ; \quad D^*_{j+1/2} - D_j = -ms \quad ; \quad e^*_j - e_j = 0 = m^*_j - m_j \tag{68}$$

$$B^*_{j+1/2} - B_j = -\frac{c}{m}\left(D^*_{j+1/2} - D_j\right) \quad . \tag{69}$$

Along the second leg:

$$\frac{dB}{ds} = 0 \quad ; \quad \frac{dD}{ds} = D \quad ; \quad \frac{de}{ds} = -e \quad ; \quad \frac{dm}{ds} = 0 \tag{70}$$

$$B_{j,2} = B^*_{j+1/2} \quad ; \quad \ln\frac{D_{j,2}}{D^*_{j+1/2}} = s = -\ln\frac{e_{j,2}}{e_j} \quad ; \quad m_{j,2} = m_j \tag{71}$$

$$e_{j,2}D_{j,2} = e_j D^*_{j+1/2} \rightarrow E^{**}_{j+1/2} = E^*_{j+1/2} \quad . \tag{72}$$

Along the third leg:

$$\frac{dB}{ds} = B \quad ; \quad \frac{dD}{ds} = 0 \quad ; \quad \frac{de}{ds} = 0 \quad ; \quad \frac{dm}{ds} = -m \tag{73}$$

$$\ln\frac{B_{j,3}}{B_{j,2}} = s = -\ln\frac{m_{j,3}}{m_{j,2}} \quad ; \quad B_{j,3} \triangleq B^{**}_{j+1/2} \tag{74}$$

$$m_{j,3}B^{**}_{j+1/2} = m_j B^*_{j+1/2} \rightarrow H^{**}_{j+1/2} = H^*_{j+1/2} \tag{75}$$

and along the final leg:

$$\frac{dB}{ds} = e \quad ; \quad \frac{dD}{ds} = c \quad ; \quad \frac{de}{ds} = 0 = \frac{dm}{ds} \quad ; \quad e_{j+1} - e_{j,3} = 0 = m_{j+1} - m_{j,3} \tag{76}$$

$$\left(B_{j+1} - B^{**}_{j+1/2}\right) = \frac{e}{c}\left(D_{j+1} - D^{**}_{j+1/2}\right) \quad . \tag{77}$$

Putting these relations together, one finds finally:

$$H^*_{j+1/2} = m_j B^*_{j+1/2} = \frac{\left[c_j B_j + e_j D_j\right] + \left[c_{j+1} B_{j+1} - e_{j+1} D_{j+1}\right]}{\left(c_j/m_j\right) + \left(c_{j+1}/m_{j+1}\right)} \tag{78a}$$

$$E^*_{j+1/2} = e_j D^*_{j+1/2} = \frac{\left[c_j D_j + m_j B_j\right] + \left[c_{j+1} D_{j+1} - m_{j+1} B_{j+1}\right]}{\left(c_j/e_j\right) + \left(c_{j+1}/e_{j+1}\right)} \quad . \tag{78b}$$

We thus have found that the appropriate fluxes involve a weighted average of the Riemann invariants on either side of the interface. The electric terms that now appear in H^*, and the magnetic terms in E^*, when these expressions are substituted into (58), result in a diffusion–like second difference term that acts to damp the integration process. These terms in fact provide the necessary stability that is missing from the simple central difference approximation.

The generalization of this approach to three dimensions is straightforward: The flux contribution along each of the three coordinate directions is computed independently, and then all three flux terms are added to give $\partial w/\partial t$ in first order. Along the $\vec{\xi}$ direction, the fluxes are:

$$\vec{\xi} \times E^* = \vec{\xi} \times \frac{\left[c^- \vec{D}^- - \hat{\xi} \times \vec{H}^-\right] + \left[c^+ \vec{D}^+ + \hat{\xi} \times \vec{H}^+\right]}{(c\epsilon)^- + (c\epsilon)^+} \tag{79a}$$

$$\vec{\xi} \times H^* = \vec{\xi} \times \frac{\left[c^- \vec{B}^- + \hat{\xi} \times \vec{E}^-\right] + \left[c^+ \vec{B}^+ - \hat{\xi} \times \vec{E}^+\right]}{(\mu c)^- + (\mu c)^+} \quad . \tag{79b}$$

which are the same as the fluxes given by equation (35) when there is no resistive sheet at the cell interface.

5.0 UNSTRUCTURED FINITE–VOLUME TREATMENT

5.1 Space/Time Discretization

The major feature of our discretization approach that distinguishes it from other finite–volume and finite–difference procedures is that the electric and magnetic field unknowns are co–located in both space and time, rather than being assigned to two interpenetrating spatial grids and separated a half–step in time. These field unknowns are the volume averages of E and H within each cell in the space–filling grid.

Staggered–grid methods automatically achieve second–order accuracy in space, while co–located field algorithms require near–neighbor corrections. However, there is a fundamental equivalence between these methods in terms of the achievable accuracy and stability of the integration process.

Both approaches typically use explicit time integration, which means that the upper limit on the allowable size of the time step Δt is determined by the physical size and shape of the smallest cells, corresponding roughly to the time that light takes to cross one of these cells. Implicit integration schemes can choose larger time steps, but they require the inversion of a banded matrix the size of the whole grid, and their ability to preserve phase information is not known.

The unstructured algorithm we have developed is applicable to any grid that fills the computational domain with polyhedral cells. In particular, we have implemented the necessary bookkeeping procedures to deal with hexahedra (such as cubes), tetrahedra, and prisms (translations of a triangle out of its plane of definition). A generalization to curved interfaces is straightforward, but not regarded as useful at this time. These three polyhedral geometries allow us to specialize the code in various ways, for instance to check against the structured–grid Lax–Wendroff algorithm in our present solver RCS3D, or to run purely two–dimensional cases.

Each polyhedron in the computational domain is specified by the location (x, y, z) of its vertices in physical space. From these locations, all the necessary geometrical quantities, including areas, surface normals, and centroidal locations are computed. As stated earlier, each field unknown attributed to a given polyhedron is considered to be an average of the field over the volume of the polyhedron. The six components of E and H at one time level are thus stored according to an index α that runs over all polyhedra. Quantities related to the polyhedral faces, such as face normals, are stored according to another index that runs over all faces.

The interior faces of a given polyhedron are kept distinct from those of the neighboring polyhedra that share faces with it in a purely geometrical sense. This allows, for instance, for any type of impedance boundary condition to be applied at the boundary between cells. Thus, each polyhedral face has a co–face with a distinct face index, and each such co–face belongs to its own polyhedron.

5.2 Polynomial Representation and Least Squares

To go beyond representation of the fields as simple volume averages, we have chosen initially to implement linear polynomial functions for both E and H. Higher order polynomial representations will follow the same general procedures. The essential question is how the higher order terms in these polynomials are to be determined from near–neighbor data, so as to achieve the desired level of accuracy within each cell. In our unstructured approach, this evaluation is closely tied to the time integration procedure through the Riemann fluxes at each interface. Ultimately, this preserves time accuracy as well as accuracy in space.

In the first step of this method, first–order Riemann fluxes are constructed at each interface of a cell from the volume–averaged fields on either side of the interface. For Maxwell's equations, these fluxes are the tangential field components just inside the boundaries of the cell. To complete the specification at the cell surface, the normal components of E and H are taken to be the normal components of the volume–averaged fields. This maintains overall charge conservation within the cell.

These boundary data are sufficient to determine all the terms in a linear polynomial fit to either E or H by a procedure such as least–squares minimization of the fitting error integrated over the boundary. If we denote the vector polynomial to be fitted as $\vec{A}$ and its boundary values as $\vec{A}^*$, then the quantity to be minimized is

$$e = \int_{\text{(cell boundary)}} \left(\vec{A} - \vec{A}^*\right)^2 dS \quad . \tag{80}$$

Taking derivatives of e with respect to each polynomial coefficient in $\vec{A}$ results in a nonsingular set of linear equations for these coefficients. For consistency, one constrains the constant term in $\vec{A}$ to be equal to the known volume average of $\vec{A}$ over the cell.

A separate set of equations is obtained for each Cartesian component of $\vec{A}$. If one writes, *e.g.*,

$$A_x(\vec{r}) = \langle A_x \rangle_\alpha + (x - x_\alpha)\frac{\partial A_x}{\partial x} + (y - y_\alpha)\frac{\partial A_x}{\partial y} + (z - z_\alpha)\frac{\partial A_x}{\partial z} \quad , \tag{81}$$

where the angular brackets denote volume averaging and *e.g.*, $x_\alpha = \langle x \rangle_\alpha$, then these equations become

$$\begin{aligned} &\int_{\partial\alpha} (x - x_\alpha)\left[(x - x_\alpha)\frac{\partial A_x}{\partial x} + (y - y_\alpha)\frac{\partial A_x}{\partial y} + (z - z_\alpha)\frac{\partial A_x}{\partial z}\right] dS \\ &\qquad = \int_{\partial\alpha} (x - x_\alpha)\left[A_x^* - \langle A_x \rangle_\alpha\right] dS \end{aligned} \tag{82a}$$

$$\begin{aligned} &\int_{\partial\alpha} (y - y_\alpha)\left[(x - x_\alpha)\frac{\partial A_x}{\partial x} + (y - y_\alpha)\frac{\partial A_x}{\partial y} + (z - z_\alpha)\frac{\partial A_x}{\partial z}\right] dS \\ &\qquad = \int_{\partial\alpha} (y - y_\alpha)\left[A_x^* - \langle A_x \rangle_\alpha\right] dS \end{aligned} \tag{82b}$$

$$\int_{\partial\alpha} (z - z_\alpha) \left[(x - x_\alpha) \frac{\partial A_x}{\partial x} + (y - y_\alpha) \frac{\partial A_x}{\partial y} + (z - z_\alpha) \frac{\partial A_x}{\partial z} \right] dS$$
$$= \int_{\partial\alpha} (z - z_\alpha) [A_x^* - \langle A_x \rangle_\alpha] \, dS \quad , \tag{82c}$$

where we have denoted the cell boundary as $\partial\alpha$. These equations can be solved by inverting the matrix M whose elements are the quadratic moments

$$M_{ij} = \int_{\partial\alpha} \hat{i} \cdot (\vec{r} - \vec{r}_\alpha) \hat{j} \cdot (\vec{r} - \vec{r}_\alpha) \, dS \quad , \tag{83}$$

where $\hat{i}$ and $\hat{j}$ are unit vectors in the respective coordinate directions.

For a linear polynomial fit, there is a simpler alternative procedure that we have implemented to evaluate these linear terms. From the divergence theorem, the average value of any derivative over the cell volume can be rewritten as a surface integral:

$$\frac{1}{V_\alpha} \int_\alpha \frac{\partial \rho}{\partial x} dV = \frac{1}{V_\alpha} \int_{\partial\alpha} \hat{n}_x \rho dS \quad , \tag{84}$$

where $\hat{n}$ is the unit outward normal on the boundary $\partial\alpha$ and V_α is the cell volume. In particular, if ρ is a linear function of $\vec{r}$ then $\partial\rho/\partial x$ is constant and equal to this volume average, which can be calculated just from the values of ρ on the boundary. For every component of A, we can replace its boundary values by the corresponding component of A^* to obtain the approximation

$$\vec{\nabla} \vec{A}_\alpha \simeq \frac{1}{V_\alpha} \int_{\partial\alpha} \hat{n} \vec{A}^* dS \triangleq K_\alpha \quad , \tag{85}$$

which is equivalent to using $\hat{n}$ as the weight in the method of weighted residuals applied to the difference $A - A^*$. The quantity K_α is a vector dyadic. Since we have chosen $\hat{n} \cdot \vec{A}^* = \hat{n} \cdot \langle \vec{A} \rangle_\alpha$, we can make use of the vector identity $\vec{a} = \hat{n} (\hat{n} \cdot \vec{a}) - \hat{n} \times (\hat{n} \times \vec{a})$ to rewrite the integral as

$$K_\alpha = \frac{1}{V_\alpha} \int_{\partial\alpha} \hat{n} \left[\hat{n} \times \{ \hat{n} \times (\langle \vec{A} \rangle_\alpha - \vec{A}^*) \} \right] dS \quad , \tag{86}$$

which will be more convenient to compute in terms of the tangential components of $\vec{A}^*$. This particular weighting can be shown to result from a variational principal that assumes each Cartesian component of $\vec{A}^*$ is the boundary value of a solution of Laplace's equation inside the cell.

5.3 The Unstructured Second–Order Algorithm

An algorithm that maintains second–order accuracy in both space and time can be constructed from the linear polynomial representation as follows:

$$\langle Q \rangle_\alpha^{m+1/2} = \langle Q \rangle_\alpha^m - \frac{\Delta t}{2V_\alpha} \int_{\partial\alpha} \hat{n} \cdot F(Q_\alpha^{*m}) \, dS \tag{87}$$

$$K_\alpha^m = \frac{1}{V_\alpha}\int_{\partial\alpha} \hat{n} Q_\alpha^{*m} dS = \frac{1}{V_\alpha}\int_{\partial\alpha} \hat{n}\left(\hat{n}\times\left\{\hat{n}\times\left[\langle Q\rangle_\alpha^m - Q_\alpha^{*m}\right]\right\}\right) dS \tag{88}$$

$$Q_\alpha^{m+1/2}(\vec{r}) = \langle Q\rangle_\alpha^{m+1/2} + (\vec{r}-\vec{r}_\alpha)\cdot K_\alpha^m \quad \text{for } \vec{r} \text{ in cell } \alpha \tag{89}$$

$$\langle Q\rangle_\alpha^{m+1} = \langle Q\rangle_\alpha^m - \frac{\Delta t}{V_\alpha}\int_{\partial\alpha} \hat{n}\cdot F\left(Q_\alpha^{*(m+1/2)}\right) dS \quad . \tag{90}$$

Here we have written Maxwell's equations symbolically as

$$\frac{\partial Q}{\partial t} + \nabla\cdot F(Q) = 0 \quad , \quad Q = \left(\vec{D},\vec{B}\right) \quad , \tag{91}$$

and the solution of the Riemann problem just inside a cell interface is denoted Q^*. This solution depends only on the values of $Q(\vec{r})$ immediately on either side of the interface. These are the cell–average values for Q^{*m} and the linear polynomial values for $Q^{*(m+1/2)}$.

6. UNSTRUCTURED GRIDDING AND RESULTS

There are three steps involved in the unstructured grid setup. 1) Geometry modeling, 2) surface grid setup, and 3) volume grid setup. Appendix A3, sections of a contract report on geometry and gridding prepared by the Computational Fluid Dynamics group at Rockwell Science Center, include a detailed writeup on the unstructured grid generation process using a procedure called the "advancing front method." Once the unstructured grid is set up (hedahedron, prism, or tetrahedron), a preprocessor is used to define all the quantities required by the unstructured grid–based Maxwewll's solver. Some of them are,

1. Cell volumes and cell face normals,
2. Face/coface connection (from this information one constructs the face and node near neighbor connection), and
3. Boundary face numbers (both body and outer boundary).

This grid related information is used in the flux summation process.

Problems in CEM involve arbitrarily shaped three–dimensional geometries that need to be represented properly in the computer simulation. In addition to the external shape, CEM also requires modeling the interior of the penetrable structure. Depending on the formulation (differential or integral), one may choose either a structured grid or an unstructured grid setup.

Two gridding issues that need to be addressed in EM computations are: 1) number of grid points per wavelength to properly represent the fields in and around a scatterer; and 2) how far should the outer boundary be placed from the scattering object to adequately simulate the nonreflecting boundary condition. In general, the number of points/wavelength is not determined by wavelength alone, and involves the body dimensions (characteristic body size with respect to wavelength) also. The outer boundary location, theoretically, can be right on the body surface itself; however, the computational implementation of nonreflecting boundary conditions requires the outer boundary at a few (2 to 5) wavelengths away from the surface. Again, if one can construct higher order accurate implementations of nonreflecting boundary conditions, the outer boundary can be brought very close to the scattering surface. In general, the necessary grid resolution is provided only around and near the body surface. Between the body and the outer boundary, the mesh is allowed to stretch resulting in very crude (3 to 5 points per wavelength) meshes near the outer boundary regions.

The free space wavelength is reduced to smaller values inside a material (as ϵ and μ become large, the speed of propagation, $c = \frac{1}{\sqrt{\epsilon\mu}}$, goes down, causing the wavelength to scale accordingly). Thus, the grid resolution must take into account material properties to adequately resolve the fields inside material zones.

The number of grid points per wavelength required depends on the order of accuracy of the numerical scheme. A second–order accurate scheme usually requires at least ten grid points per local wavelength. One may be able to use a higher order scheme and minimize the number of grid points. However, as the order of accuracy goes up, the scheme will also

require more computations per grid point, which may offset the execution savings with fewer grid points.

The requirement that the fields are resolved accurately with proper grid resolution makes CEM problems computationally intensive, requiring large scale supercomputing. For example, to compute the radar cross section of a typical aircraft at 1 GHz, even if one used 10 grid cells per wavelength, it will require tens of millions of grid points.

Some of the salient features of the current CEM capabilities are

- Time–domain Maxwell's equations
- Proven algorithms from CFD
- Single pulse (multiple frequency, transient) or continuous incident wave (single frequency, time harmonic steady state)
- Numerical grid generation — structured multizone grid or unstructured grid

  Complex geometry with layered radar absorbing media
- Lossy or lossless material properties

  Frequency and time dependent properties

  Thin structures (resistive card, lossy paint)
- Vector/parallel code architecture — 2 GFLOPS demonstrated on the Cray–YMP with 8 processors, and 10 GFLOPS on the Cray–C90 with 16 processors. Scalable performance demonstrated on both the nCUBE and the Intel Paragon.

  Received the 1990 CRAY Gigaflop Performance Award

  Received the 1993 Computerworld Smithsonian Award
- Pre– and post–processor graphics/animation
- Application to scattering (RCS), radiation (antenna), EMP/EMI/EMC, and bioelectromagnetics problems
- Ideal for CFD/CEM optimization studies.

The CEM code has been extensively tested for the following geometries.

1) Canonical objects such as spheres, cylinders, ogives, thin rods, cones, airfoils, and a circular disc
2) Almond shaped target
3) Inlets of various shapes (square, circular, curved, $\cdots$) including the presence of infinite ground plane
4) Flat plates of various planforms
5) Double sphere
6) Complete wing geometries with layers

7) Finned projectile and cone–cylinder combinations

8) Scattering from ship–like targets

9) Complete fighter targets

Some sample results are shown here to illustrate the present capability. Figure 6.1 shows an unstructured grid set up for an almond. Figure 6.2 shows an unstructured grid–based CEM computation for a complete fighter. Figure 6.3 shows a bioelectromagnetic computation involving the aborption of microwave radiation by a human used in the hyperthermia treatment of cancer. Figure 6.4 shows results for a monopole antenna. Figure 6.5 shows results for a two– dimensional photonic band gap structure used as a filtering device. Currently, this CEM technology is being applied to study the interference of pylon mountings in the field test RCS measurements of low observable platforms of interest to RATSCAT shown in Figure 6.6.

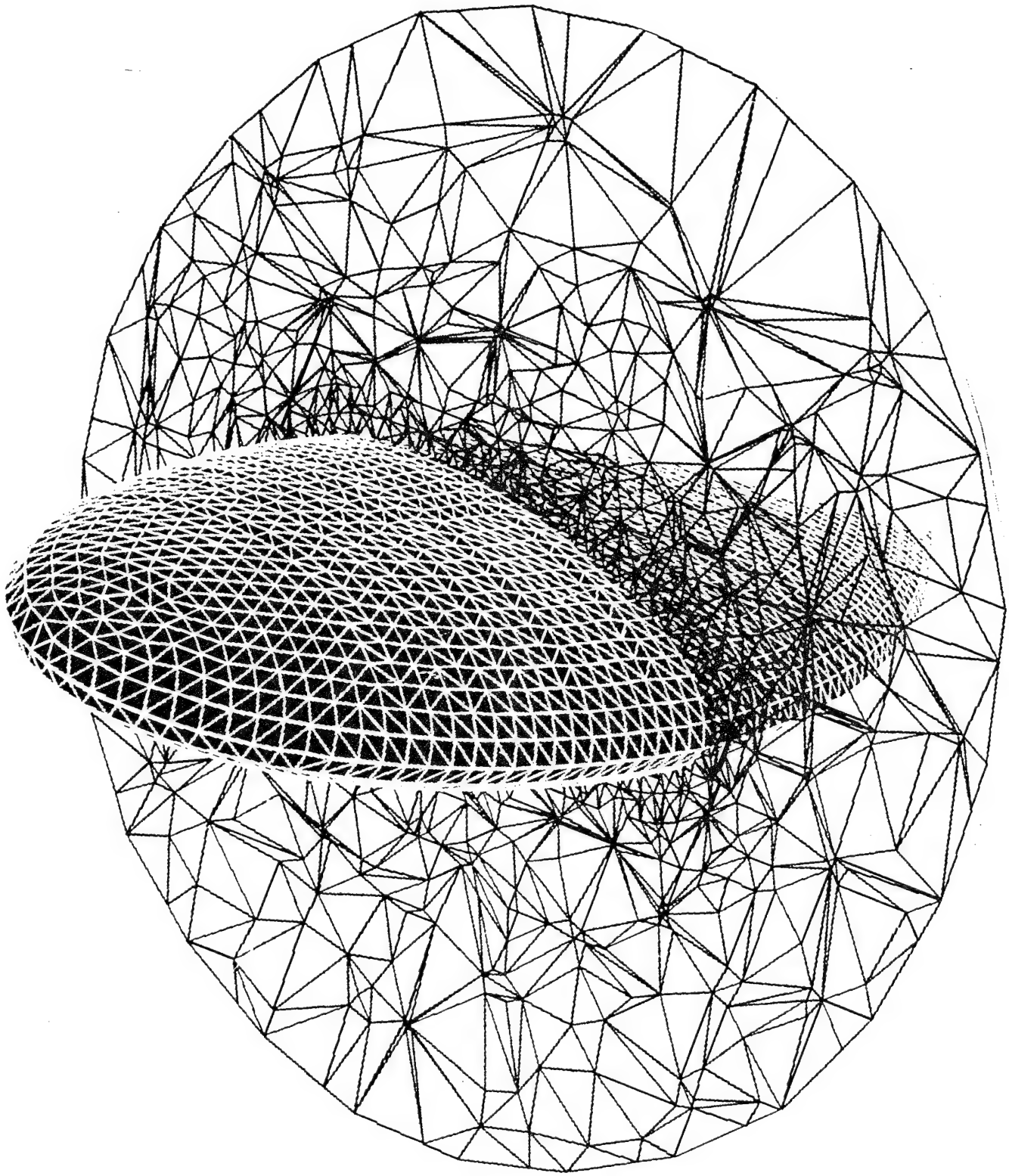


Figure 6.1 Unstructured gridding for an almond

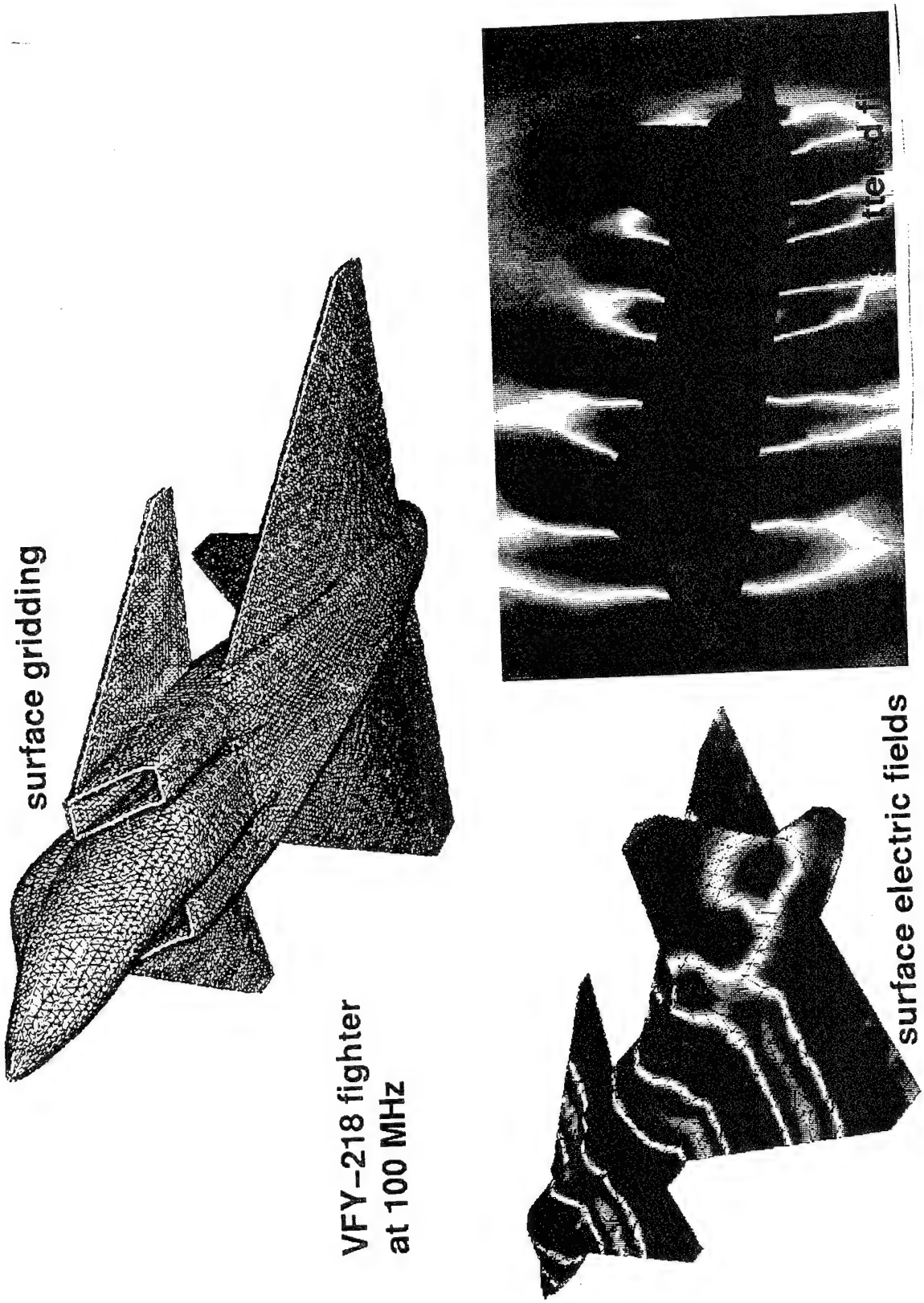


Figure 6.2 Unstructured grid–based CEM for a complete fighter

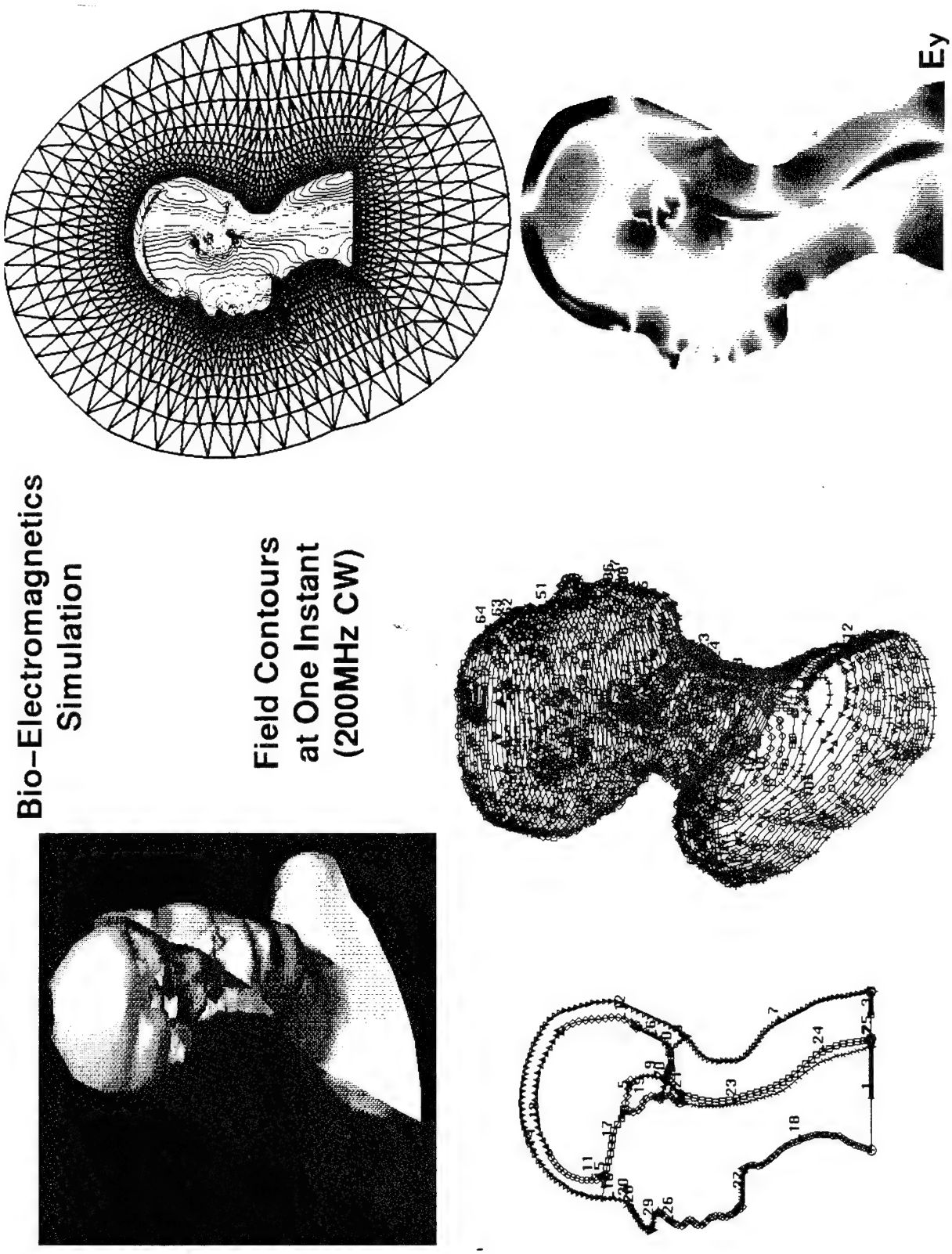


Figure 6.3 Bioelectromagnetics application of CEM

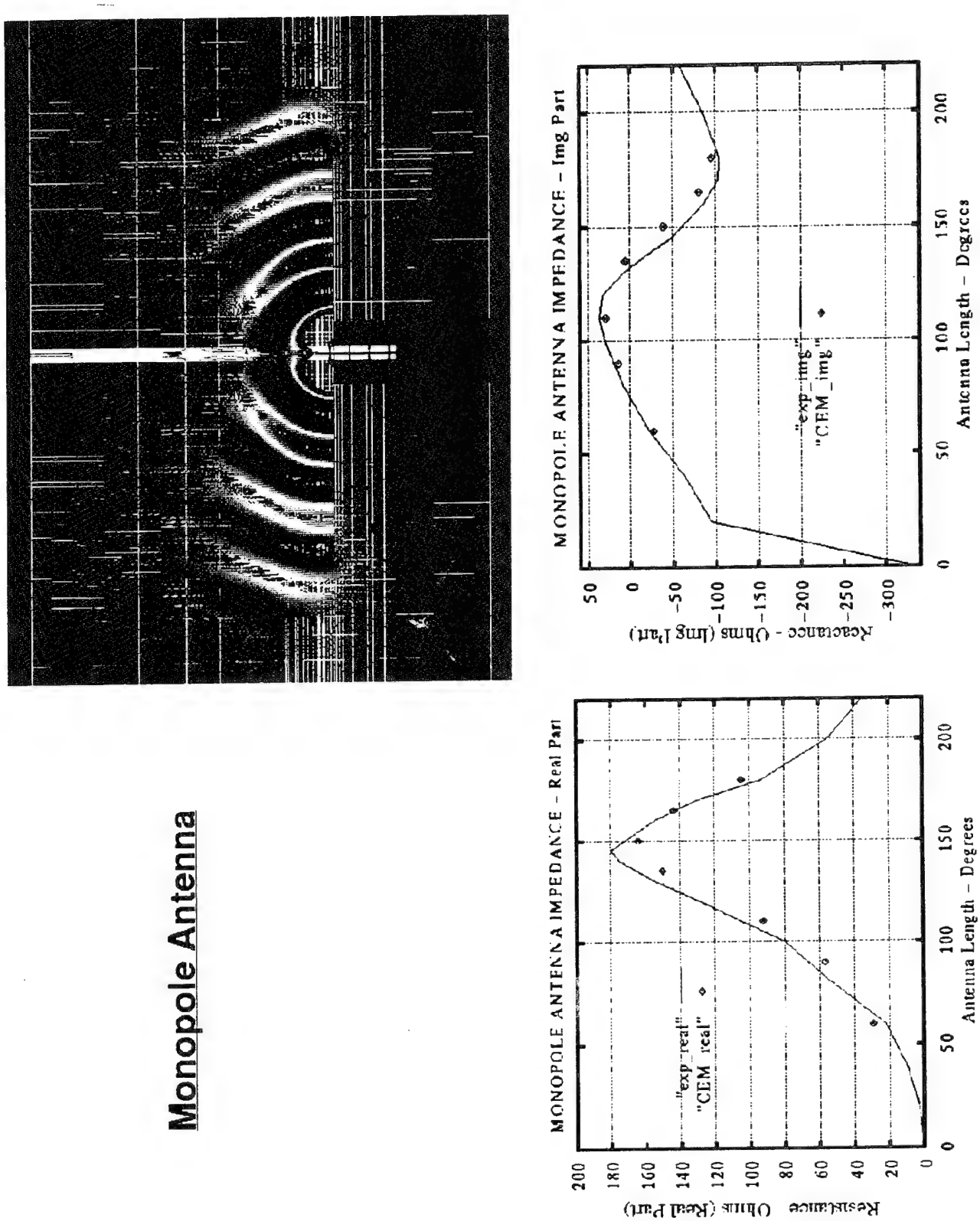


Figure 6.4 Simulation of monopole antenna radiation

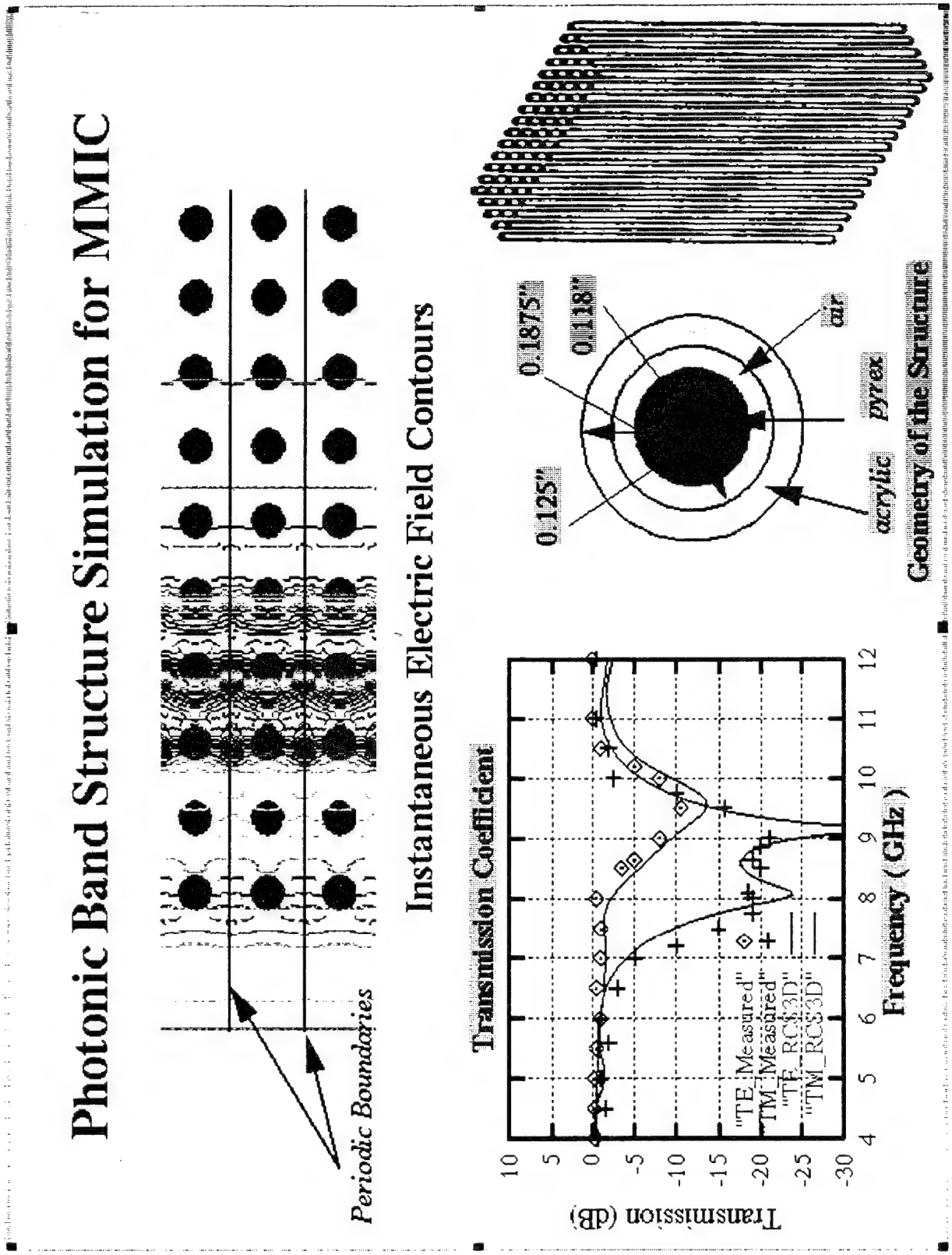


Figure 6.5 Photonic band gap periodic structure

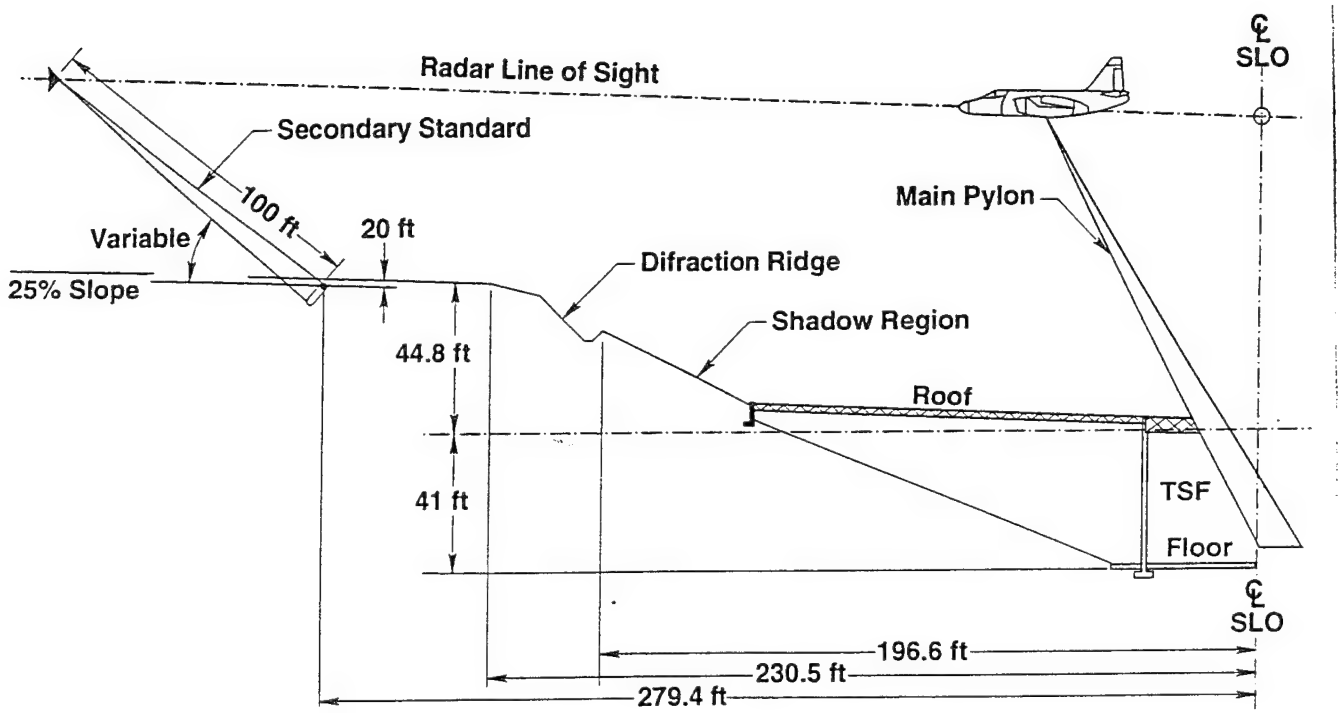


Figure 6.6 Application of CEM for studying interference of pylon mountings in RCS measurements

7.0 MASSIVELY PARALLEL COMPUTING

Parallel Implementation

With the emergence of massively parallel computing architectures with potential for teraflops performance, any code development activity must effectively utilize the computer architecture in achieving the proper load balance with minimum internodal data communication.

The structured finite–volume code was originally developed and optimized for vector computer architectures. The implementation of the code on a distributed memory parallel architecture was accomplished by re-using much of the original vector code. Additional coding was added for handling inter-processor communication and other functions unique to the parallel implementation.

Parallelization Strategy

For the structured formulation of the finite–volume code the computational domain surrounding the target geometry is composed of 3-dimensional 6 sided volumes of grid points called zones or blocks. Each side or face of a zone either connects to another zone or has a boundary condition defined on that face (perfect conducting surface, outer boundary, etc.). The parallel algorithm takes advantage of this multi-zonal gridding capability in order to divide work among processors. The various zones are grouped onto processors, with each processor obtaining a solution for the cells within its own local set of zones.

Communication Requirements

The solution procedure does not allow for processors to proceed completely asynchronously. Solving for cells on zone faces that are connected to other zones requires information from within the adjacent zone. This information may be available locally if the adjacent zone resides on the same processor, or message passing may be required if the adjacent zone resides on another processor. This boundary update message passing or flux transfer message passing is done twice per solution time step and forms the bulk of the parallel code's message passing requirements.

Load Balancing

Load balancing is achieved by mapping zones onto processors. Perfect load balancing requires that each processor have the same number of zones, each containing the same number of grid points and equal numbers and types of boundary condition cells. Simple geometries may usually be zoned in such a manner as to obtain perfect load balancing. For complex geometries perfect load balancing is much more difficult, but adequate load balancing may usually be obtained by mapping a close to equal number of grid points onto each processor.

Scalability Results

Validation and timing studies have been performed on a 512-node nCUBE and a 208-node Intel Paragon. Currently the code shows good scalability on evenly balanced test cases. These cases typically had simple gridding requirements and a straight forward domain decomposition. The results show that inter-processor communication due to flux transfer never becomes a dominant time factor even on problems with large numbers of

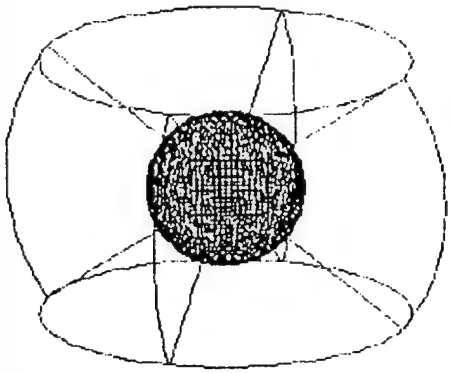
grid points run on many processors. The sphere test case illustrated in Figure 7.1 shows how problem size and number of processors can be increased while solution time remains level. Perfectly conducting sphere grids were run on 6, 24, and 96 processors of the Intel Paragon. The number of grid points per processor remained constant at approximately 60,000 resulting in total grid sizes of approximately 0.35, 1.4, and 5.7 million grid points for the three cases. Since increasing the number of processors results in an increased number of zonal interfaces, flux message passing requirements increase throughout the system. Despite this increase in required message passing, communication times did not change appreciably.

Complex problems such as full scale fighter geometries also show encouraging results. Figure 7.2 shows timing results and zoning for the VFY218 fighter gridded for a frequency of 500MHz with a 10 point per wavelength resolution. A total of 58 zones and 2.2 million grid points were required. The grid was run on an Intel Paragon using 28, 61, and 128 processors. Preliminary timing data reveals that communication overhead remains at between 1 and 2.5 percent of the total solve time and that solution speedup occurs as the problem is distributed over more processors. Speedup may be improved by addressing load balancing issues arising from complex zoning arrangements.

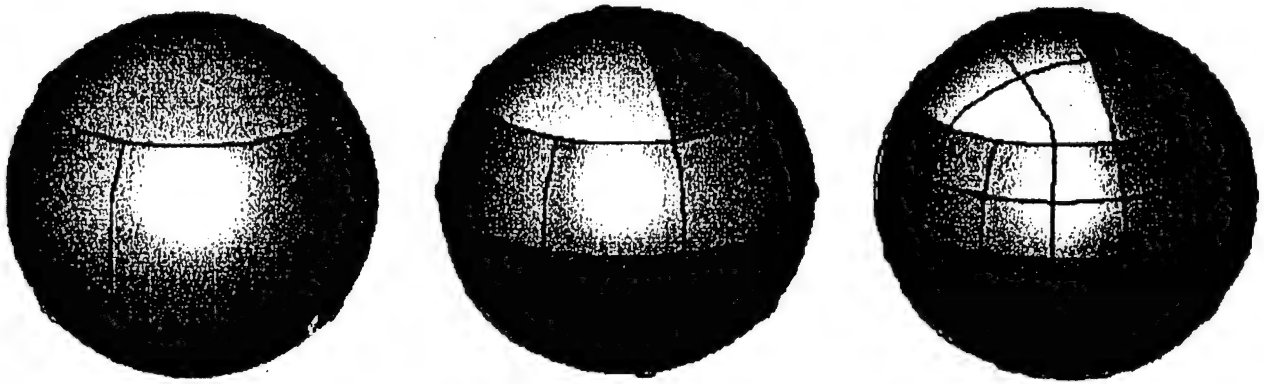
Parallel CEM for a Sphere

Domain Decomposition

6 Zone Volume Blocking



Surface Zone Outlines



6 Node 24 Node 96 Node

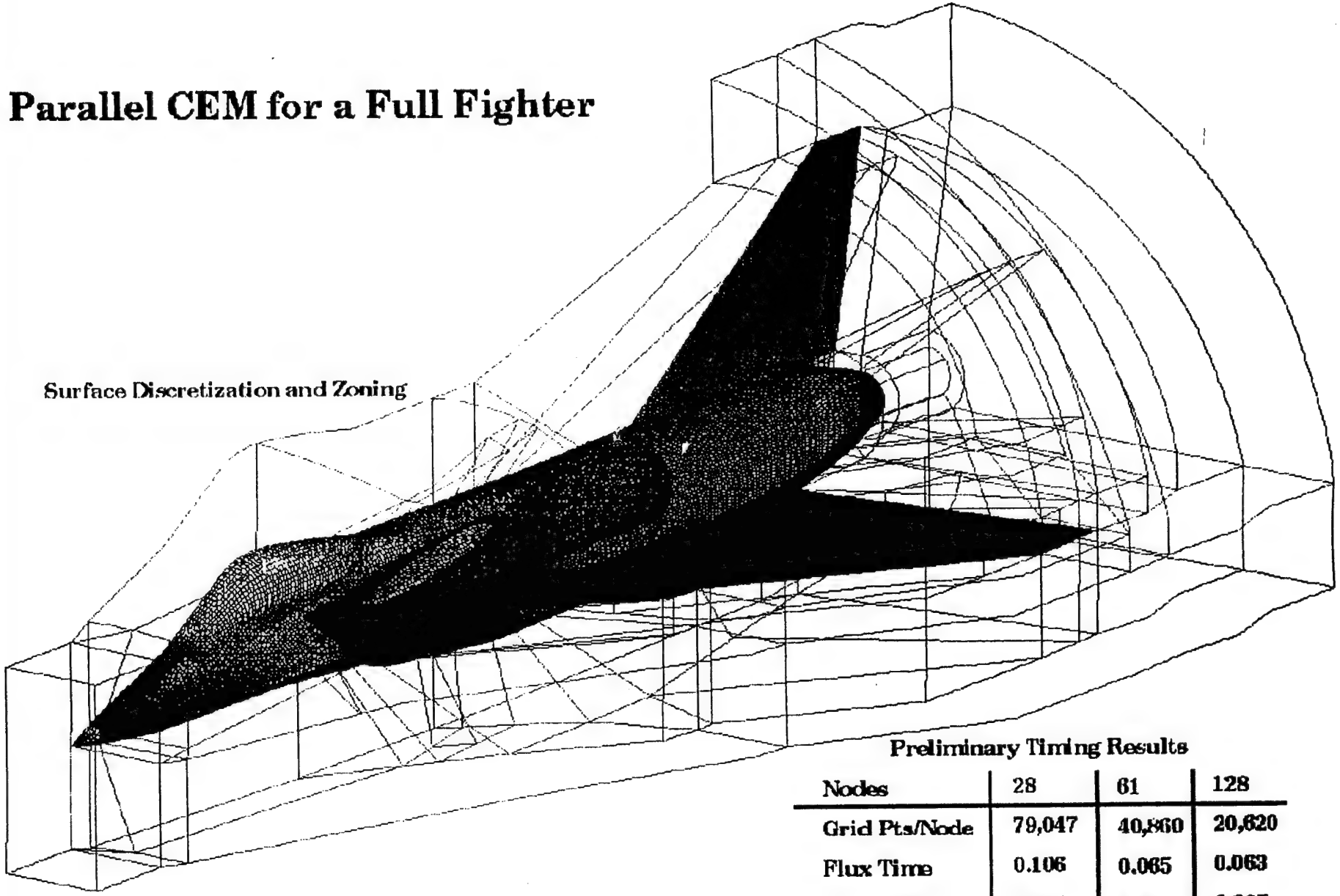
Timing Results

Nodes	6	24	96
Grid Pts/Node	59,049	59,049	59,059
Total Grid Pts	354,294	1,417,176	5,668,704
Flux Time	0.044	0.052	0.0349
Solve Time	4.515	4.518	4.509

(Time – seconds per time step)

Figure 7.1 Scaling of computer resources with problem size for a sphere

Parallel CEM for a Full Fighter



Preliminary Timing Results

Nodes	28	61	128
Grid Pts/Node	79,047	40,860	20,620
Flux Time	0.106	0.065	0.063
Solve Time	8.324	4.433	2.605

(Time – seconds per time step)

Figure 7.2 Scaling of computer resources with problem size for a fighter

8. FURTHER DEVELOPMENT OF THE UNSTRUCTURED–GRID CEM CODE

Based on the preliminary results obtained so far from the unstructured–grid CEM code, many issues need further study to mature this technology comparable to the structured grid RCS3D code. Some of them are,

1. The process of constructing a polynomial basis function within each cell for the electric and magnetic fields requires further study for determining the accuracy. Current procedure involves the use of Riemann fluxes at cell interfaces and greatly simplifies the process of evaluating the slope information. For arbitrary cell arrangements this procedure requires further work, especially at boundary faces.

2. We need to look at higher order basis functions greater than second order accuracy. There is a computational penalty that goes with the evaluation of higher order polynomials. We need to understand the balance between computational efficiency and accuracy.

3. Current procedure does not solve the Maxwell equations at a metal boundary (option 10 type procedure in RCS3D) and updates the electric and magnetic fields only at cell centroids as a cell averaged value. We need to look at ways to improve the accuracy, especially for traveling waves, by solving Maxwell's equations right at metal boundaries within the unstructured grid approach (option 1 in RCS3D).

4. The current code can have only one type of a cell arrangemnt in the entire computational domain (hexahedron, prism, or tetrahedron). For geometric flexibility we need to upgrade the pilot code to include combinations of different cell shapes.

5. The present algorithm reduces to Lax–Wendroff for hexahedral cell shapes. This allows one to construct a hybrid structured/unstructured grid solver to exploit the virtues of both approaches.

6. RCS3D has many types of boundary conditions such as impedance layers, resistive cards, and can handle frequency (dispersive) and time dependent (active) material properties. The unstructured CEM needs further development to include all these features.

7. Pre and post processing for unstructured arrangement needs further work. The FAST capability developed by NASA Ames Research Center is rather limited and cannot handle hexahedron, prism, and tetrahedron combinations. Also, it cannot handle plotting of electric and magnetic fields at cell centroids while the grid is specified at nodes.

8. The use of tetrahedra can result in small cell volumes (compared to a similar hexahedron). This can cause the time step size to be small (maybe a factor of four less compared to hexahedra cells). We need to look at pointwise implicit schemes to allow larger time steps while maintaining stability.

9. The current "advancing front" grid generator needs further work to include constraints on minimal allowable cell volumes. In the present approach, arbitrarily small volumes can result at a distance from the scattering object causing small time steps.

10. We need further study to understand the grid resolution requirements in the unstructured tetrahedral arrangements (how many cells per wavelength?).

11. Unstructured algorithms are inherently not well suited for optimum vectorization because of indirect addressing. For Cray-like machines (coarse grain vector/parallel), we need to understand the proper data and code structure to acieve the type of MFLOP ratings demonstrated by RCS3D. The current version runs about three times slower than RCS3D.

12. The real power of an unstructured grid approach is in using MIMD architectures. We need to understand how to achieve proper load balancing and choices for domain decomposition. This is a crucial research topic for demonstrating large RCS computations such as a complete fighter at giga Hertz frequencies.

9. REFERENCES

1. V. Shankar, "Research to Application – Supercomputing Trends for the 90's and Opportunities for Interdisciplinary Computations," AIAA Dryden Research Lecture, AIAA Paper NO. 91–0002, 29th Aerospace Sciences Meeting, January 7–10, 1991, Reno, Nevada.

2. A. Harten and S.R. Chakravarthy, "Multi–dimensional ENO schemes for general geometries," UCLA Computational and Applied Mathematics (CAM) Report 91–16, September 1991; also ICASE Report 91–76, September 1991.

3. S.R. Chakravarthy and S. Osher, "Computing with High–Resolution Upwind Schemes for Hyperbolic Equations," Lectures in Applied Mathematics **22** (1985).

4. Sukumar R. Chakravarthy, "Development of Upwind Schemes for the Euler Equations," NASA Contractor Report 4043, January 1987.

5. V. Shankar, "A Unified Full Potential Scheme for Subsonic, Transonic, and Supersonic Flows," AIAA Paper No. 85–1643, AIAA 18th Fluid Dynamics, Plasmadynamics, and Lasers Conference, Cincinnati, July 16–18, 1985.

6. V. Shankar, K.–Y. Szema, and S. Osher, "A Conservative Type–Dependent Full Potential Method for the Treatment of Supersonic Flows with Embedded Subsonic Regions," AIAA Journal **25 (2)** (1987).

7. V. Shankar, H. Ide, J. Gorski, and S. Osher, "A Fast, Time–Accurate Unsteady Full Potential Scheme," AIAA Journal **25 (2)** (1987).

8. V. Shankar and H. Ide, "Unsteady Full Potential Computations for Complex Configurations," AIAA Paper No. 87–0110, AIAA 25th Aerospace Sciences Meeting, Reno, January 1987.

9. K.–Y. Szema, S.R. Chakravarthy, and H. Dresser, "Multizone Euler Marching Technique for Flows over Multibody Configurations," AIAA Paper No. 87–0592, January 1987, and AIAA Paper No. 88–0276, January 1988.

10. S.R. Chakravarthy, K.–Y. Szema, and J.W. Haney, "Unified Nose–to–Tail Computational Method for Hypersonic Vehicle Applications," AIAA Paper No. 88–2564, June 1988.

11. S.R. Chakravarthy and K.–Y. Szema, "Advances in Finite Difference Techniques for Computational Fluid Dynamics," Chapter 1, "State–of–the–Art Surveys on Computational Mechanics," Eds. A.K. Noor and J.T. Oden, American Society of Mechanical Engineers, 1989.

12. S.R. Chakravarthy, "Some Aspects of Essentially Non-Oscillatory (ENO) Formulations for the Euler Equations," NASA Contractor Report 4285, May 1990.

13. D. Pan and S. Chakravarthy, "Unified Formulation for Incompressible Flows," AIAA Paper 89–0122.

14. P. Lax, "Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves," SIAM, Philadelphia, 1973.

15. V. Shankar, W.F. Hall, and A.H. Mohammadian, "A time domain differential solver for electromagnetic scattering problems," Proc. IEEE 77 (1989), 709.

16. V. Shankar, W.F. Hall, and A.H. Mohammadian, "A CFD–based finite–volume procedure for computational electromagnetics — interdisciplinary applications of CFD methods," in Proc. AIAA 9th Computational Fluid Dynamics Conference, Buffalo, 13–15 June 1989.

17. A.H. Mohammadian, V. Shankar, and W.F. Hall, "Application of the time–domain differential solver to 2–D electromagnetic penetration problems, IEEE Trans. Magn. 25 (1989), 2875.

18. V. Shankar, W.F. Hall, and A.H. Mohammadian, "A three–dimensional Maxwell's equation solver for computation of scattering from layered media," IEEE Trans. Magn. 25 (1989), 3098.

19. V. Shankar, W.F. Hall, A.H. Mohammadian, and S. Chakravarthy, "Application of CFD based methods to problems in computational science," Comput. Syst. Eng. 1 (1990), 7.

20. V. Shankar, W.F. Hall, and A.H. Mohammadian, "A time domain, finite–volume treatment for the Maxwell equations," Electromagnetics 10 (1990), 127.

21. A.H. Mohammadian, V. Shankar, and W.F. Hall, "Computation of electromagnetic scattering and radiation using a time–domain finite–volume discretization procedure," Computer Phys. Comm. 68 (1991), 175–196.

22. V. Shankar, "CFD spinoff — computational electromagnetics for radar cross section (RCS) studies," AIAA Paper No. 90–3055, AIAA 8th Applied Aerodynamics Conference, Portland, August 20–22, 1990.

23. V. Shankar, "Gigaflop performance algorithms in computational science — application to Maxwell's equations for electromagnetics computations," AIAA 10th Computational Fluid Dynamics Conference, Session No. CFD–10, June 24–27, 1991, Honolulu.

24. V. Shankar, "Radar cross section of a finned projectile," Final Report for Contract DAAD05–90–P–9348, U.S. Army Ballistic Research Laboratory, November 1991.

25. V. Shankar, "Progress report for CFD–based CEM code development," NASA Contract N62269–90–C–0257, November 1991.

26. R.F. Warming and R.M. Beam, "Upwind Second Order Difference Schemes and Applications in Aerodynamic Flows," AIAA J., Vol. 14, No. 9, September 1976.

27. D. G. Bishop and D. A. Anderson, "A Comparison of Finite–Volume Time–Domain Schemes for the Maxwell Equations," AIAA Paper No. 92–0456, 30th Aerospace Sciences Meeting, January 6–9, 1992, Reno.

28. G. Mur, "Absorbing Boundary Conditions for the Finite–Difference Approximation of the Time–Domain Electromagnetic–Field Equations," IEEE Transactions on Electromagnetic Compatibility, Vol. EMC–23, No. 4, pp 377–382, November, 1981.

29. B. Engquist and A. Majda, "Absorbing Boundary Conditions for the Numerical Simulation of Waves," Math. Comp., Vol. 31, pp. 629–651, July 1977.

30. V. Shankar, W. F. Hall, A. Mohammadian, and C. Rowell, "Development of a Finite–Volume, Time–Domain Solver for Maxwell's Equations," A final report prepared for NASA/NADC under contract N6269–90–C–0257, May 93.

31. S. Osher and S. Chakravarthy, "High Resolution Schemes and the Entropy Condition," SIAM J. Numerical Analysis, Vol. 21, No. 5, pp 955–984, October, 1984.

A1. Treatment for Impedance Layer

I. INTRODUCTION

The electromagnetic signatures of perfectly conducting objects may be reduced by coating their surface with a thin layer of lossy dielectric, absorbing material such as paint. This, together with the shape factor, are the basic tools for developing low observable structures. In computing the radar cross section (RCS) of these objects, if the brute force approach is attempted, the problem is often too large to be handled, since effective coatings usually have high complex permeabilities and permittivities. Therefore, some form of approximate boundary conditions[35] have to be used to replace the dielectric layer by an impedance surface at the interface between the dielectric and free space.

The impedance boundary condition, however, is a frequency domain concept. In this appendix, it is shown how this concept can be extended to the time domain and implemented for the finite–volume time–domain (FVTD) method.

II. FORMULATION

The most basic problem to which the impedance boundary condition can be applied is that of a plane wave incident upon a material half–space. Provided that the magnitude of the complex refractive index of material is large, i.e., $|N| >> 1$ where $N = \sqrt{\mu\epsilon/\mu_0\epsilon_0}$ with μ, ϵ, μ_0, and ϵ_0 being the complex permeabilities and permittivities of the material half–space and free space, respectively, then the material interface may be replaced by an impedance surface located at the interface on which the electric and magnetic fields satisfy the following condition, known as the impedance boundary condition:

$$\hat{n} \times (\hat{n} \times \vec{E}) = -\eta \hat{n} \times \vec{H} \qquad (A1.1)$$

where η is the characteristic impedance of the material medium and $\hat{n}$ is a unit vector normal to the interface in the free space. This impedance boundary condition remains valid when the interface is a curved surface provided the additional constraint $|\mathrm{Im}(N)|k_0\rho >> 1$ is imposed. ρ is the minimum radius of the curvature.

This basic concept may be simply extended to a multilayered medium. Only the simple case of a lossy material coating over a conducting body will be discussed here. From a simple transmission line analogy, the coated object may be replaced by a surface whose impedance is given by:

$$\eta = -iZ \tan(k_0 d\sqrt{\mu\epsilon}) \qquad (A1.2)$$

where Z is the characteristic impedance of the material coating, d is its thickness, k_0 is the propagation constant in free space, and a time dependence $exp(-i\omega t)$ has been assumed.

In the framework of the FVTD method, Maxwell's equations in the partial differential form in a body–fitted coordinate system are numerically integrated in the time and over the faces of each grid cell to compute the field values inside that cell, as explained in the

previous sections. When one or more of the faces of a cell are impedance surfaces, then a special treatment is necessary. The tangential components of the electric and magnetic fields of each cell used in the time integration are derived from jump conditions usually defined in the time domain. In the present case, however, it advantageous to start with jump conditions in the frequency domain and derive the expressions for the tangential fields on the impedance faces of the cells involved. Then, the time–domain tangential fields are derived from their frequency–domain counterparts through Fourier transformations.

Figure A1.1 shows a cell in the computational grid. The tangential fields for this cell with one of its ξ–constant faces being an impedance surface will be derived. Derivation for η– and ζ–constant faces are identical. The jump condition across the positive characteristic as depicted in Figure A1.1 may be expressed as:

$$Z^+\left(\vec{H}^+ - \vec{H}^*\right) = \hat{\xi} \times \left(\vec{E}^+ - \vec{E}^*\right) \qquad (A1.3a)$$

where the field quantities bearing superscript '+' are defined at the centroid, those with asterisk on the faces of the cell, and Z^+ is the characteristic impedance of the material filling the cell. Equation (A1.3a) basically relates the discontinuities of the electric and magnetic fields across the eigenvalues $\lambda^+ = c|\xi|$ of Maxwell's equations where c is the speed of light in the medium. For both the scattered and total fields, (A1.3a) assumes the same form. However, it is meant to present scattered fields here. In terms of the scattered fields, (A1.1) may be written as follows:

$$\hat{\xi} \times \left[\hat{\xi} \times \left(\vec{E}^* + \vec{E}^i\right)\right] = -\eta\hat{\xi} \times \left(\vec{H}^* + \vec{H}^i\right) \qquad (A1.3b)$$

where η is given by (A1.2). From (A1.3) $\hat{\xi} \times \vec{H}^*$ is obtained.

$$\hat{\xi} \times \vec{H}^* = \frac{\hat{\xi} \times \left(\vec{W}^+ - \vec{W}^i\right)}{\eta + Z^+} \qquad (A1.4)$$

where $\vec{W}^{i,+} = Z^+\vec{H}^{i,+} - \hat{\xi} \times \vec{E}^{i,+}$. It is advantageous, however, to find $\hat{\xi} \times \vec{E}^*$ from a jump condition similar to (A1.3a) in the time domain. This will be done later. Inverse Fourier transformation of (A1.4) results in the desired expression for the tangential magnetic field on the impedance face of the cell in the time domain which is as follows:

$$\begin{aligned}\hat{\xi} \times \vec{\mathcal{H}}^*(t) = &\int_0^t F(t-\tau)\hat{\xi} \times \vec{\mathcal{W}}^+(\tau)d\tau \\ &+ \int_0^t F(t-\tau)\hat{\xi} \times \vec{\mathcal{W}}^i(\tau)d\tau - \hat{\xi} \times \vec{\mathcal{H}}^i(t)\end{aligned} \qquad (A1.5)$$

where:

$$
\begin{aligned}
F(t) &= \mathcal{F}^{-1}\left(\frac{1}{Z^{+}+\eta}\right) \quad , \\
\vec{\mathcal{W}} &= \mathcal{F}^{-1}(\vec{W}) \quad , \\
Re[\eta] &= \frac{\delta N_1(\omega)+\gamma N_2(\omega)}{D(\omega)} \quad , \\
Im[\eta] &= \frac{\gamma N_1(\omega)+\delta N_2(\omega)}{D(\omega)} \quad , \\
D(\omega) &= 1+\tan^2(k_0 d\alpha)\tanh^2(k_0 d\beta) \quad , \\
N_1(\omega) &= \tan(k_0 d\alpha)/\cosh^2(k_0 d\beta) \quad , \\
N_2(\omega) &= \tanh(k_0 d\beta)/\cos^2(k_0 d\alpha) \quad , \\
\alpha &= \pm\sqrt{\frac{1}{2}\left(Re[\mu\epsilon]+|\mu\epsilon|\right)} \quad , \\
\beta &= \pm\sqrt{\frac{1}{2}\left(-Re[\mu\epsilon]+|\mu\epsilon|\right)} \quad , \\
\gamma &= \pm\sqrt{\frac{1}{2}\left(Re\left[\frac{\mu}{\epsilon}\right]+\left|\frac{\mu}{\epsilon}\right|\right)} \quad , \\
\delta &= \pm\sqrt{\frac{1}{2}\left(-Re\left[\frac{\mu}{\epsilon}\right]+\left|\frac{\mu}{\epsilon}\right|\right)} \quad .
\end{aligned}
$$

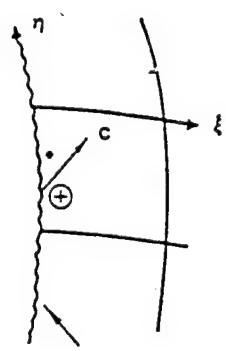


Fig. A1.1. A Cell with Impedance Face.

The plus signs apply to a wave traveling in the positive coordinate direction, while the minus signs to a wave traveling in the negative coordinate direction.

For a time–harmonic or narrow–band plane wave excitation, provided the steady state solution is desired, a fictitious dynamics for $\eta(\omega)$ will be constructed whose inverse Fourier transform can be evaluated in a closed form. The following two cases will be considered here:

1) $\frac{Z^+ + Re[\eta]}{Im[\eta]} < 0$

In this case we let:

$$\eta(\omega) = r - i\omega x \tag{A1.6}$$

where r and x are constant over the frequency band of interest. The straightforward application of an inverse Fourier transform results in the desired function in the time domain

$$F(t) = ae^{-bt} \tag{A1.7a}$$

where:

$$a = \frac{1}{x} \tag{A1.7b}$$

$$b = \frac{r + Z^+}{x} \quad . \tag{A1.7c}$$

2) $\frac{Z^- + Re[\eta]}{Im[\eta]} > 0$

$\eta(\omega)$ in this case will be considered to be:

$$\eta(\omega) = r' - i\frac{x'}{\omega} \quad . \tag{A1.8}$$

The inverse Fourier transform of (A1.8) results in the following function:

$$F(t) = c\delta(t) + ae^{-bt} \tag{A1.9a}$$

where:

$$a = \frac{x'}{Z^+ + r'} \tag{A1.9b}$$

$$b = -\frac{x'}{Z^+ + r'} \tag{A1.9c}$$

$$c = \frac{1}{r' + Z^+} \quad . \tag{A1.9d}$$

For notation simplification we use (A1.9a) for both cases with $c = 0$ for Case 1.

A plane, time–harmonic incident field may be represented by:

$$\vec{\mathcal{E}}_i(t) = \hat{\theta}\cos(\vec{k}\cdot\vec{R} - \omega t) \tag{A1.10a}$$

$$\vec{\mathcal{H}}_i(t) = \hat{\phi}\frac{1}{Z^+}\cos(\vec{k}\cdot\vec{R} - \omega t) \quad . \tag{A1.10b}$$

From (A1.10), $\vec{W}^i(t)$ may be constructed:

$$\begin{aligned}\vec{W}^i(t) &= Z^+\vec{\mathcal{H}}^i(t) - \hat{\xi}\times\vec{\mathcal{E}}^i(t)\\ &= \hat{\omega}\cos(\vec{K}\cdot\vec{R} - \omega t)\end{aligned} \tag{A1.11}$$

where $\widehat{W} = \hat{\phi} + \hat{\xi} \times \hat{\theta}$.

Equations (A1.9), (A1.10), and (A1.11) are substituted in (A1.5) to obtain the magnetic field flux term in the time domain for the impedance surface:

$$\begin{aligned} \hat{\xi} \times \vec{\mathcal{H}}^*(t) &= c\hat{\xi} \times \vec{W}^+(t) + c\hat{\xi} \times \vec{W}^i(t) \\ &+ a \int_0^t e^{-b(t-\tau)} \hat{\xi} \times \vec{W}^+(\tau) d\tau \\ &+ a \int_0^t e^{-b(t-\tau)} \hat{\xi} \times \vec{W}^i(\tau) d\tau - \hat{\xi} \times \vec{\mathcal{H}}^i(t) \end{aligned} \quad . \qquad (A1.12)$$

The first integral in the right hand side of (A1.12) will result in a recursive algorithm. The details follow. Let:

$$\vec{\mathcal{T}}(t) = \int_0^t e^{-b(t-\tau)} \hat{\xi} \times \vec{W}^+(\tau) d\tau \quad . \qquad (A1.13)$$

$\vec{\mathcal{T}}(t)$ may be regarded as a formal solution to the following first order differential equation:

$$\frac{d\vec{\mathcal{T}}}{dt} + b\vec{\mathcal{T}} = \hat{\xi} \times \vec{W}^+(\tau) \quad . \qquad (A1.14)$$

It is found that the following implicit algorithm can provide accurate solution to $\vec{\mathcal{T}}$ for a large range of values of b:

$$\frac{\vec{\mathcal{T}}(t+\Delta t) - \vec{\mathcal{T}}(t)}{\Delta t} + \frac{b}{2}\left[\vec{\mathcal{T}}(t+\Delta t) + \vec{\mathcal{T}}(t)\right] = \frac{1}{2}\left[\hat{\xi} \times \vec{W}^+(t+\Delta t) + \hat{\xi} \times \vec{W}^+(t)\right] \quad . \qquad (A1.15)$$

This will result in the following recursive formula for computing $\vec{\mathcal{T}}(t)$ for the discrete time variable:

$$\vec{\mathcal{T}}_n = \frac{2 - b\Delta t}{2 + b\Delta t} \vec{\mathcal{T}}_{n-1} + \frac{\Delta t}{2 + b\Delta t} \left[\left(\hat{\xi} \times \vec{W}^+\right)^n + \left(\hat{\xi} \times \vec{W}^+\right)^{n-1} \right] \qquad (A1.16)$$

where:

$$\begin{aligned} \vec{\mathcal{T}}_n &= \vec{\mathcal{T}}(n\Delta t) \\ \vec{\mathcal{T}}_0 &= 0 \\ \left(\hat{\xi} \times \vec{W}^+\right)^n &= \hat{\xi} \times \vec{W}^+(n\Delta t) \end{aligned} \quad .$$

The second integral in (A1.12) can be evaluated in a closed form. Let:

$$\begin{aligned} \vec{U}(t) &= \int_0^t e^{-b(t-\tau)} \hat{\xi} \times \vec{W}^i(\tau) d\tau \\ &= \hat{\xi} \times \widehat{W} \int_0^t e^{-b(t-\tau)} \cos(\vec{k} \cdot \vec{R} - W\tau) d\tau \end{aligned} \quad . \qquad (A1.17)$$

Then it may be shown that:

$$\vec{U}(t) = \hat{\xi} \times \widehat{W} \frac{e^{-bt}}{\sqrt{b^2+\omega^2}} \left[e^{b(\tau+\phi)} \cos(\vec{k} \cdot \vec{R} - \omega\tau) \right] \Bigg|_{\tau=-\phi}^{\tau=t-\phi}$$
$$= \frac{e^{-bt}}{\sqrt{b^2+\omega^2}} \left[e^{b(\tau+\phi)} \hat{\xi} \times \vec{W}^i(\tau) \right] \Bigg|_{\tau=-\phi}^{\tau=t-\phi} \quad . \qquad (A1.18)$$

Expressions (A1.16) and (A1.18) are substituted in (A1.12) to obtain the desired magnetic field flux:

$$\hat{\xi} \times \vec{\mathcal{H}}^*(t) = c\hat{\xi} \times \vec{W}^+(t) + c\hat{\xi} \times \vec{W}^i(t) + a\vec{T}(t) + a\vec{U}(t) - \hat{\xi} \times \vec{\mathcal{H}}^i(t) \quad . \qquad (A1.19)$$

The results in (A1.19) may be presented using the discretized time variable n:

$$\left(\hat{\xi} \times \vec{\mathcal{H}}^*\right)^n = c\left(\hat{\xi} \times \vec{W}^+\right)^n + c\left(\hat{\xi} \times \vec{W}^i\right)^n + a\vec{T}_n + a\vec{U}_n - \left(\hat{\xi} \times \vec{\mathcal{H}}\right)^i \qquad (A1.20)$$

where $\vec{U}_n = \vec{U}(n\Delta t)$.

The electric field flux, $\hat{\xi} \times \vec{\mathcal{E}}^*$, may be obtained directly from the cross product of the unit vector $\hat{\xi}$ and the following jump condition in the time domain:

$$\vec{\mathcal{E}}^* - \vec{\mathcal{E}}^+ = \hat{\xi} \times \left[Z^+(\vec{\mathcal{H}}^* - \vec{\mathcal{H}}^+) \right] \quad . \qquad (A1.21)$$

A2. Treatment for Anisotropic Media

Radar absorbing materials, such as filled honeycomb structures, may exhibit considerable anisotropy in their response to electromagnetic fields. Treating these structures as continuous media requires tensor definitions for ϵ and μ. Thus, the interface fluxes derived earlier for isotropic media need to be generalized. We start from the conservation–law form of Maxwell's equations in curvilinear coordinates (ξ, η, ζ), equation (32):

$$\frac{\partial}{\partial t}\left(\frac{Q}{\mathcal{J}}\right) + \frac{\partial}{\partial \xi}\begin{pmatrix} \vec{\xi}\times\vec{E}/\mathcal{J} \\ -\vec{\xi}\times\vec{H}/\mathcal{J}\end{pmatrix} + \frac{\partial}{\partial \eta}\begin{pmatrix} \vec{\eta}\times\vec{E}/\mathcal{J} \\ -\vec{\eta}\times\vec{H}/\mathcal{J}\end{pmatrix} + \frac{\partial}{\partial \zeta}\begin{pmatrix} \vec{\zeta}\times\vec{E}/\mathcal{J} \\ -\vec{\zeta}\times\vec{H}/\mathcal{J}\end{pmatrix} = \frac{S}{\mathcal{J}} \quad ,$$

where $Q = (\vec{B}, \vec{D})$ and $S = (0, -\vec{J})$. As in the isotropic case, we shall solve the one–dimensional Riemann problem associated with one coordinate direction, and we consider that the finite–volume cell is bounded by surfaces of constant ξ, η, and ζ.

Writing $\vec{E} = \epsilon^{-1}\vec{D}$ and $\vec{H} = \mu^{-1}\vec{B}$, one finds, for example, that the ξ–derivative term above takes the form:

$$\frac{\partial}{\partial \xi}\begin{pmatrix} \vec{\xi}\times\vec{E}/\mathcal{J} \\ -\vec{\xi}\times\vec{H}/\mathcal{J}\end{pmatrix} = \frac{\partial}{\partial \xi}\left(\begin{bmatrix} 0, \vec{\xi}\times\epsilon^{-1} \\ -\vec{\xi}\times\mu^{-1}, 0\end{bmatrix}\frac{Q}{\mathcal{J}}\right) \triangleq \frac{\partial}{\partial \xi}\left(A\frac{Q}{\mathcal{J}}\right) \quad , \qquad (A2.1)$$

where the position dependence of ϵ and μ is accounted for in the 6×6 matrix A. In our finite–volume formalism, ϵ and μ are taken to be constant inside each cell, and the fields $\vec{B}$ and $\vec{D}$ are evaluated at the cell centroid.

Waves that propagate along the $\vec{\xi}$ direction correspond to eigenvectors of A. As we shall see, these eigenvectors are pairs $\left(\vec{B}_\alpha, \vec{E}_\alpha\right)$ having both $\vec{B}_\alpha$ and $\vec{E}_\alpha$ lying in the plane perpendicular to $\vec{\xi}$ at the interface between neighboring cells. Because the material properties may vary from cell to cell, these eigenvectors in general also change across the interface.

In terms of the 3×3 matrix $S = \left(\vec{\xi}\times\mu^{-1}\right)\left(\vec{\xi}\times\epsilon^{-1}\right)$, the right eigenvectors r_α and eigenvalues λ_α on one side of a ξ–interface can be derived in the form:

$$\begin{aligned} r_1 &= \begin{pmatrix}\hat{v}_1 \\ Z_2\hat{v}_2\end{pmatrix} ; r_2 = \begin{pmatrix}\hat{v}_2 \\ -Z_1\hat{v}_1\end{pmatrix} ; r_3 = \begin{pmatrix}\hat{v}_1 \\ -Z_2\hat{v}_2\end{pmatrix} ; \\ r_4 &= \begin{pmatrix}\hat{v}_2 \\ Z_1\hat{v}_1\end{pmatrix} ; r_5 = \begin{pmatrix}\epsilon\hat{\xi} \\ 0\end{pmatrix} ; r_6 = \begin{pmatrix}0 \\ \mu\hat{\xi}\end{pmatrix} \end{aligned} \qquad (A2.2a)$$

$$\lambda_1 = -\lambda_3 = \sqrt{-c_1 + \sqrt{c_1^2 - c_2}}\,;\ \lambda_2 = -\lambda_4 = \sqrt{-c_1 - \sqrt{c_1^2 - c_2}}\,;\ \lambda_5 = \lambda_6 = 0 \quad , \qquad (A2.2b)$$

$$c_1 = \frac{1}{2}\left(S_{11} + S_{22} + S_{33}\right) ;\ c_2 = -S_{12}S_{21} - S_{13}S_{31} - S_{23}S_{32} + S_{11}S_{22} + S_{22}S_{33} + S_{11}S_{33} \quad , \qquad (A2.2c)$$

$$Z_1 = \sqrt{(\hat{v}_2\cdot\epsilon^{-1}\hat{v}_2) + (\hat{v}_1\cdot\mu^{-1}\hat{v}_1)}\,;\ Z_2 = \sqrt{(\hat{v}_1\cdot\epsilon^{-1}\hat{v}_1)\,/\,(\hat{v}_2\cdot\mu^{-1}\hat{v}_2)} \quad . \qquad (A2.2d)$$

The physical unit vectors $\hat{v}_1$ and $\hat{v}_2$ are perpendicular to $\hat{\xi}$, and they are determined by the requirements:

$$\hat{v}_1 \cdot \mu^{-1}\hat{v}_2 = \hat{v}_2 \cdot \mu^{-1}\hat{v}_1 = \hat{v}_2 \cdot \epsilon^{-1}\hat{v}_1 = \hat{v}_1 \cdot \epsilon^{-1}\hat{v}_2 = 0 \quad . \tag{A2.3}$$

The eigenvectors r_5 and r_6 do not correspond to propagating waves because $\lambda_5 = \lambda_6 = 0$, and they are not required in constructing the interface flux.

As in the isotropic case, the basic relations constraining the interface flux are that the difference between the centroidal values of $\vec{B}$ and $\vec{D}$ and their values at the interface must lie along the eigenvectors r_α ($\alpha = 1$ to 4). Coupled with the condition that the components of $\vec{E}$ and $\vec{H}$ tangential to the interface must be the same on either side, these relations determine the following expressions for the interface fluxes:

$$\hat{\xi} \times E^* = \frac{-\hat{\eta}(\vec{a}^+ + \vec{a}^-) \cdot [\vec{H}^+ - \vec{H}^- + \vec{N} \times \vec{\xi}] - \hat{\zeta}(\vec{b}^+ + \vec{b}^-)[\vec{H}^+ - \vec{H}^- + N \times \vec{\xi}]}{\hat{\xi} \cdot (\vec{a}^+ + \vec{a}^-) \times (\vec{b}^+ + \vec{b}^-)} \quad , \tag{A2.4a}$$

$$\hat{\xi} \times H^* = \frac{\hat{\eta}(Z_1^+ Z_2^+ \vec{a}^+ + Z_1^- Z_2^- \vec{a}^-) \cdot [E^+ - E^- - M \times \hat{\xi}] + \hat{\zeta}(Z_1^+ Z_2^+ \vec{b}^+ + Z_1^- Z_2^- \vec{b}^-) \cdot [E^+ - E^- - M \times \hat{\xi}]}{\hat{\xi} \cdot (Z_1^+ Z_2^+ \vec{a}^+ + X_1^- Z_2^- \vec{a}^-) \times (Z_1^+ Z_2^+ \vec{b}^+ + Z_1^- Z_2^- \vec{b}^-)} \tag{A2.4b}$$

where the vectors $\vec{a}$ and $\vec{b}$ are given by:

$$\vec{a} = \frac{Z_1(\hat{v}_1 \cdot \hat{\eta})\hat{v}_1 + Z_2(\hat{v}_2 \cdot \hat{\eta})\hat{v}_2}{Z_1 Z_2 \hat{\xi} \cdot (\hat{v}_1 \times \hat{V}_2)} \; ; \; \vec{b} = \frac{Z_1(\hat{v}_1 \cdot \hat{\zeta})\hat{v}_1 + Z_2(\hat{v}_2 \cdot \hat{\zeta})\hat{v}_2}{Z_1 Z_2 \hat{\xi} \cdot (\hat{v}_1 \times \hat{v}_2)} \tag{A2.5}$$

and the vectors $\vec{M}$ and $\vec{N}$ are:

$$\vec{M} = Z_1^+ Z_2^+ \left[\vec{a}^+(\vec{H}^+ \cdot \hat{\eta}) + \vec{b}^+(\vec{H}^+ \cdot \hat{\zeta})\right] + Z_1^- Z_2^- \left[\vec{a}^-(\vec{H}^- \cdot \hat{\eta}) + \vec{b}^-(\vec{H} \cdot \hat{\zeta})\right] \tag{A2.6a}$$

$$\vec{N} = \left[\vec{a}^+(\vec{E}^+ \cdot \hat{\eta}) + \vec{b}^+(\vec{E}^+ \cdot \hat{\zeta})\right] + \left[\vec{a}^-(\vec{E}^- \cdot \hat{\eta}) + \vec{b}^-(\vec{E}^- \cdot \hat{\zeta})\right] \quad . \tag{A2.7a}$$

Appendix A3. Geometry and Gridding Techniques

This appendix contains Sections 12–14 of a Navy contract report, NAWCWPNS TP 8220, prepared by the Computational Fluid Dynamics group at Rockwell Science Center. The following page numbers have been taken directly from that report.

12.0 GEOMETRY FORMULATION DETAILS 117

12.1 Local node numbers and coordinates 117

12.2 Shape function 117

12.2.1 Local faces and vertices 121

12.2.2 Cell-face metrics and normals 121

12.3 Cell-face quadrature points 122

12.3.1 Special cases 123

12.4 Cell edges 123

12.5 Calculation of cell volume 124

12.6 Strong-conservation-law form in general coordinates 124

12.7 Integral and strong-conservation-law forms 124

12.8 Preservation of uniform flow 124

13.0 MESH GENERATION 140

13.1 Data structure 142

13.1.1 List for the front face 143

13.1.2 List for points in the neighborhood 144

13.1.3 Linked list between point and faces 145

13.2 Check for intersecting faces 146

13.3 Grid generation for viscous flow 146

14.0 TOPOLOGY TREATMENT 163

14.1 Node locations and nodes of cell 163

14.2 Elimination of redundant nodes and cells 163

14.3 Cells of given node 164

14.4 Common faces 164

14.5 Cell types 164
14.6 Cell division 165
14.7 Link removal 165
14.8 Octree sort and search 166

12.0 GEOMETRY FORMULATION DETAILS

The geometry formulation in UNIVERSE-series unstructured grid codes is very flexible and, in general, can handle any type of element (conservation cell). In particular, we have included three types of cells at present. These are 1) the hexahedral cell, 2) the triangular prism cell, and 3) the tetrahedral cell. The details of the geometry treatment for these three cell types are given in this section.

12.1 Local Node Numbers and Coordinates

A cell type is defined by the number and placement of its vertices. A complete definition of a cell type would also include, in addition to the vertices, any other nodes needed to define the variation of the geometric variables x, y, z in the cell (see Fig. 4.2 on higher-order elements). Taken together, the vertex nodes and other nodes can simply be referred to as the nodes of the cell.

We assume a local coordinate system for each cell (ξ, η, σ, Fig. 4.1). The vertex node numbers are defined in terms of an ordered set of integers and the values of the local coordinates at each vertex are defined for each cell type in Figs. 12.1, 12.2 and 12.3.

12.2 Shape Function

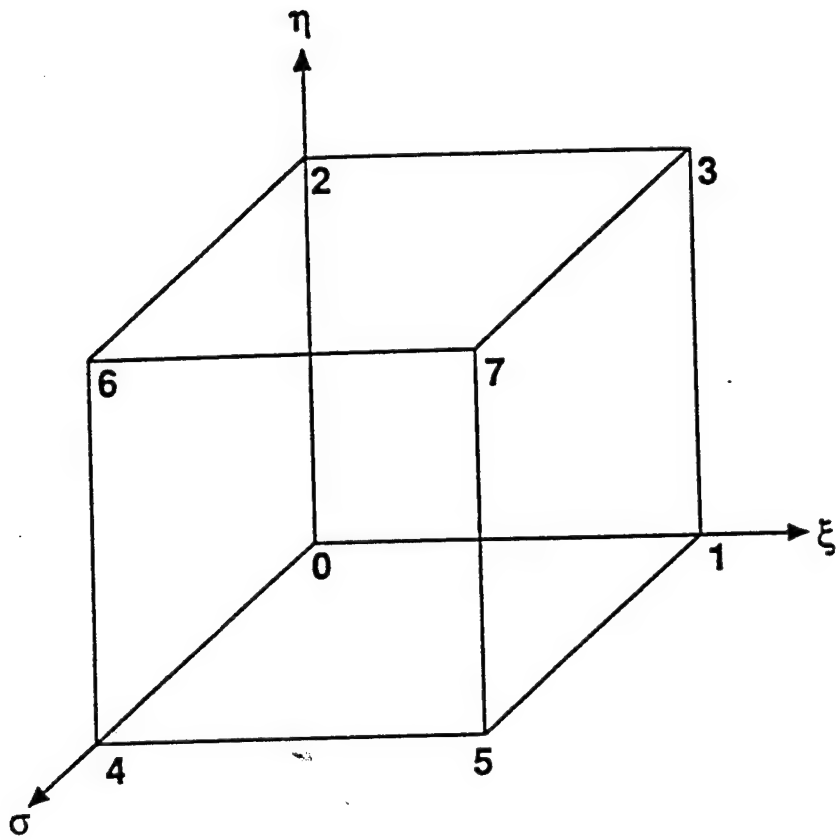
The geometric "shape function" $N_i(\xi, \eta, \sigma)$ was defined in Eqs. 4.9-11 for the three cell types. Each cell node is associated with its own shape function which is a multidimensional polynomial in ξ, η, σ:

$$N_i(\xi, \eta, \sigma) = \sum_{k=0}^{K} p_{ik} \xi^{l(k)} \eta^{m(k)} \sigma^{n(k)} \tag{12.1}$$

The polynomial coefficients can easily be obtained by solving the K equations resulting from setting

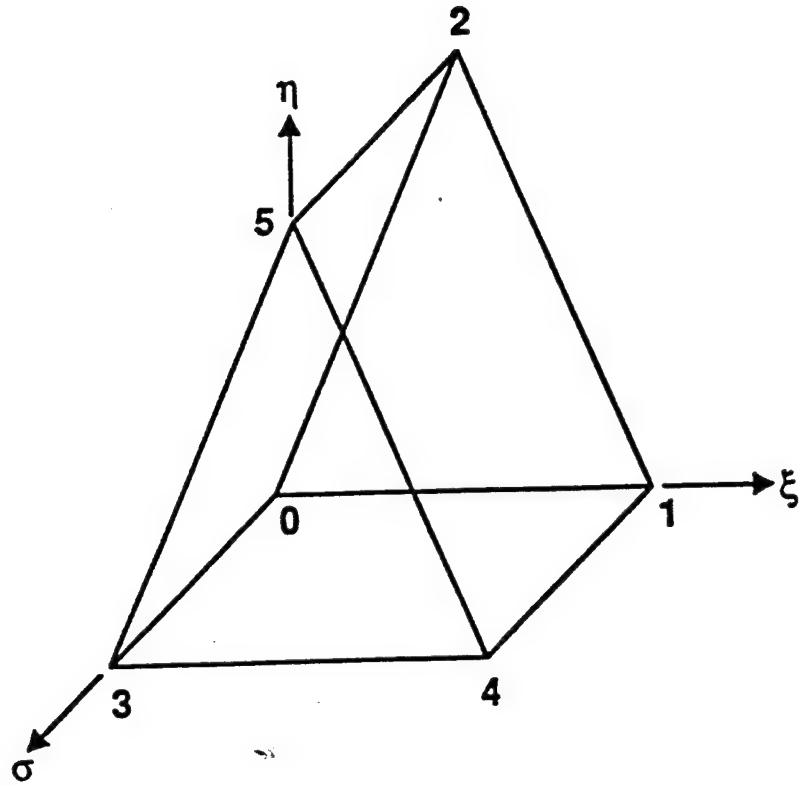
$$N_i(\xi_i, \eta_i, \sigma_i) = 1, \tag{12.2a}$$

$$N_i(\xi_j, \eta_j, \sigma_j) = 0, \quad j \neq i. \tag{12.2a}$$



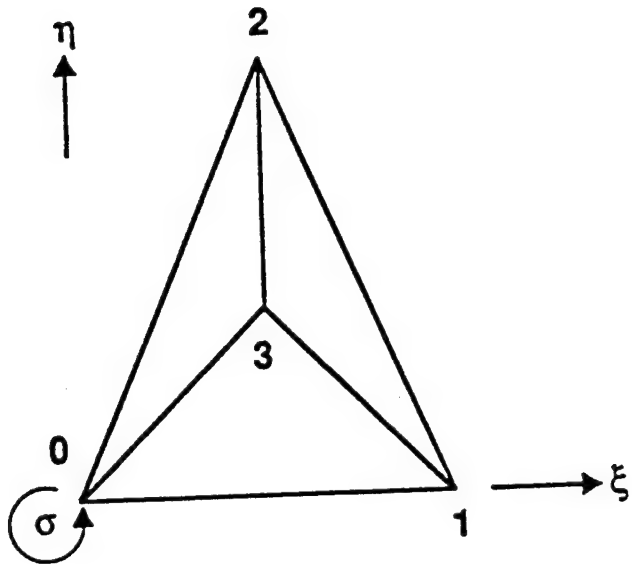
VERTEX	ξ	η	σ
0	−1	−1	−1
1	+1	−1	−1
2	−1	+1	−1
3	+1	+1	−1
4	−1	−1	+1
5	+1	−1	+1
6	−1	+1	+1
7	+1	+1	+1

Figure 12.1 Vertex nodes and coordinates for hexahedron



Vertex	ξ	η	σ
0	-1	-1	-1
1	+1	-1	-1
2	0	+1	-1
3	-1	-1	+1
4	+1	-1	+1
5	0	+1	+1

Figure 12.2 Vertex nodes and coordinates for triangular prism



Vertex	ξ	η	σ
0	-1	-1	-1
1	+1	-1	-1
2	0	1	-1
3	0	0	+1

Figure 12.3 Vertex nodes and coordinates for tetrahedron

When the three-dimensional geometry polynomial is built by forming the product of two or more lower-dimensional polynomials (Eq. 4.10, Eq. 4.11), the three-dimensional shape functions can also be constructed from the appropriate lower-dimensional shape functions.

12.2.1 Local Faces and Vertices

Each face of a cell is defined by which vertices belong to it (in terms of local node numbers). The faces and their corresponding vertices are identified for each of the three cell types in Figs. 12.4, 12.5 and 12.6.

12.2.2 Cell-Face Metrics and Normals

The method used to compute cell-face normals was outlined in Section 4 in the paragraphs following Eq. 4.11. A more detailed description is given below.

Given the geometry polynomials (Eq. 4.9, Eq. 4.10, or Eq. 4.11), the tangents to the local coordinate directions can be defined by

$$\begin{aligned} \vec{\xi} &= x_\xi \hat{j} + y_\xi \hat{k} + z_\xi \hat{l} \\ \vec{\eta} &= x_\eta \hat{j} + y_\eta \hat{k} + z_\eta \hat{l} \\ \vec{\sigma} &= x_\sigma \hat{j} + y_\sigma \hat{k} + z_\sigma \hat{l} \end{aligned} \tag{12.3}$$

Next we identify two vectors in the plane of the cell face by choosing suitable linear combinations of the above.

$$\begin{pmatrix} \vec{V}_1 \\ \vec{V}_2 \end{pmatrix} = \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \end{pmatrix} \begin{pmatrix} \vec{\xi} \\ \vec{\eta} \\ \vec{\sigma} \end{pmatrix} \tag{12.4}$$

The transformation matrix in the above can be defined as T

$$T = \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \end{pmatrix} \tag{12.5}$$

and is defined for each face of each cell type in Figs. 12.7, 12.8 and 12.9. The T matrix elements are selected appropriately to account for the fact that ξ, η, σ range,

in each cell, from -1 to $+1$ in many cases. The cross product of $\vec{V}_1$ and $\vec{V}_2$ defines the cell-face normal.

12.3 Cell-Face Quadrature Points

At every cell face, the integral of any quantity Q over that face can be replaced, up to the desired order of accuracy, by a suitable quadrature formula

$$\int_t \int_s Q(s,t)\, ds\, dt = \sum_{\text{quads.}} C_i\, Q(s_i, t_i) \tag{12.6}$$

where s, t are the representative running orthogonal coordinates tangential to the face, and subscript i denotes a quadrature point. The coefficient C_i is the "weight" assigned to the i-th quadrature point. For the three cell types being considered, one encounters only two types of cell faces — "square" or "triangular" (in terms of the running coordinates s and t, as well as the local coordinates ξ, η, σ). Tables 12.1 and 12.2 represent four-point Gaussian quadrature points and weights for such "square" and "triangular" cells. This leads to fourth-order accuracy in the evaluation of cell-face integrals. Figure 12.10 presents these quadrature points diagrammatically (not necessarily to true scale).

s	t	C
−0.577350269189626	−0.577350269189626	0.5
+0.577350269189626	−0.577350269189626	0.5
−0.577350269189626	+0.577350269189626	0.5
+0.577350269189626	+0.577350269189626	0.5

Table 12.1 Gaussian quadrature points for "square" face

s	t	C
−0.487831473351	−0.689897894859	0.318041443825
+0.487831473351	−0.689897894859	0.318041443825
−0.204988807440	+0.289898127317	0.181958556175
+0.204988807440	+0.289898127317	0.181958556175

Table 12.2 Gaussian quadrature points for "triangular" face

12.3.1 Special Cases

While, in general, the four-point Gaussian quadrature is employed for three-dimensional problems, simpler formulae (with fewer quadrature points) can be used sometimes without any loss of accuracy. For example, with hexahedral cells, the midpoint rule is sufficient for one-dimensional problems in the one significant direction being considered. For two-dimensional problems, a two-point quadrature is sufficient for faces spanning the third direction (the two points are along the line that bifurcates the cell in the third direction).

12.4 Cell edges

The need to determine the intersection between interior surface geometry elements and elements that define a cutting plane or surface for postprocessing purposes required a numbering system for cell edges to be added to the book-keeping framework. Figures 12.11, 12.12 and 12.13 illustrate the vertex nodes and cell edges for the three types of cells considered in this report.

12.5 Calculation of Cell Volume

The cell volume is related to Eqs. 4.18 and 4.19

$$V = \int\int\int_V \vec{\nabla} \cdot (x\hat{j} + y\hat{k} + z\hat{l})\, dV \tag{12.7}$$

corresponding to $i = 0$ in Eq. 4.18. The volume integration can be replaced by surface integration.

$$\begin{aligned} V &= \int\int\int_V (\vec{\nabla} \cdot \vec{X}_0)\, dV \\ &= \int\int_S (\vec{X}_0 \cdot \hat{n})\, dS \end{aligned} \tag{12.8}$$

The surface integral can, in turn, be converted to a quadrature formula based on Eq. 12.6.

12.6 Strong-Conservation-Law Form in General Coordinates

In Section 5, the strong-conservation-law form was introduced for hyperbolic systems of conservation laws in Cartesian coordinates. In future editions of this report, the invariance of that form under coordinate transformations will be discussed.

12.7 Integral and Strong-Conservation-Law Forms

There is a clear and useful geometric analogy between the integral form and the strong-conservation-law form of the equations. This will be elucidated in future editions of this report.

12.8 Preservation of Uniform Flow

Physically, uniform flow remains uniform. Numerically, this simple fact may not be true. In other words, when a numerical method is applied to update the solution from time step to time step with uniform flow as the starting solution, on nonuniform grids only a careful construction of the algorithm will guarantee that the numerical solution does not introduce spurious disturbances into uniform flow. All that is required is that formulae of high enough order of accuracy are employed in the boundary quadrature procedure. These issues will be considered in detail in future editions of this report.

Face	Vertex 0	Vertex 1	Vertex 2
0	0	1	2
1	0	1	3
2	1	2	3
3	0	2	3

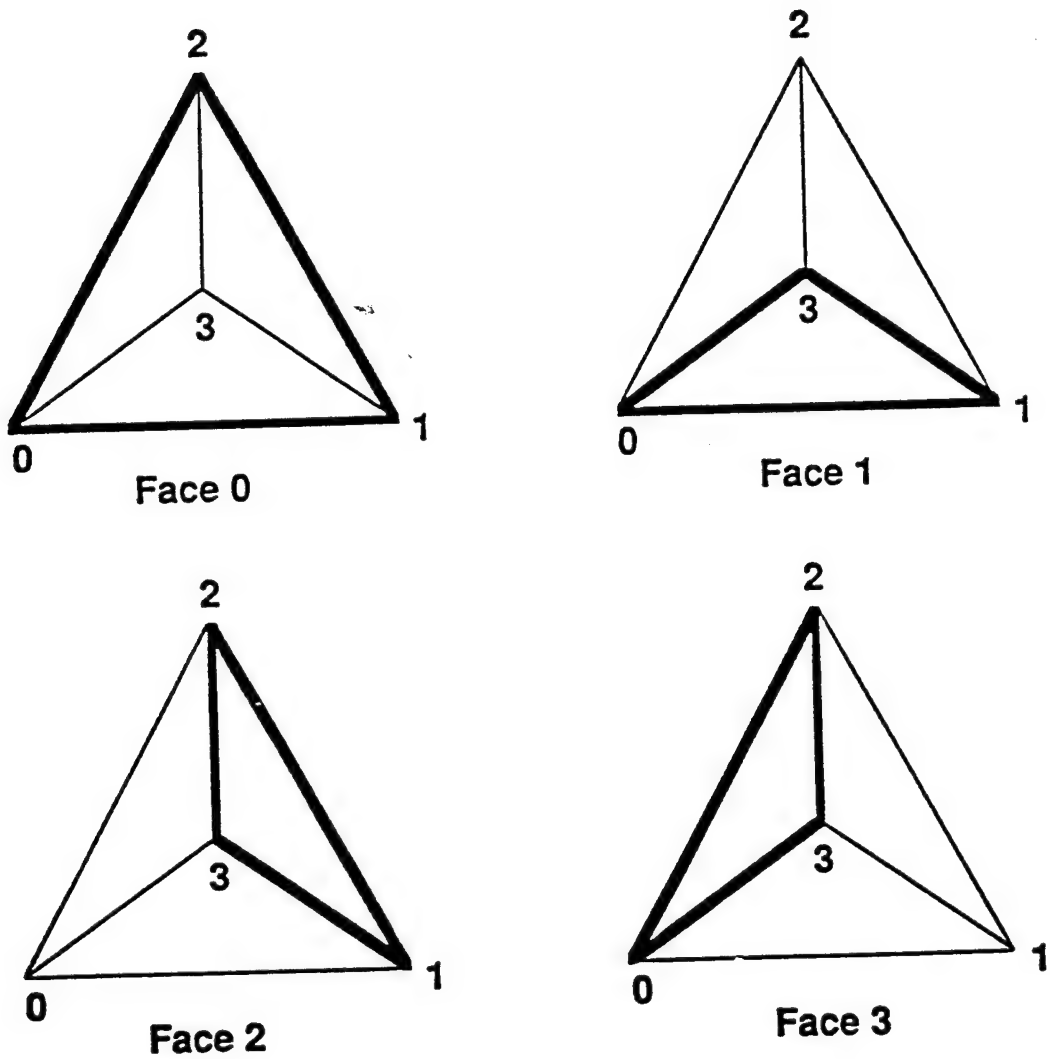
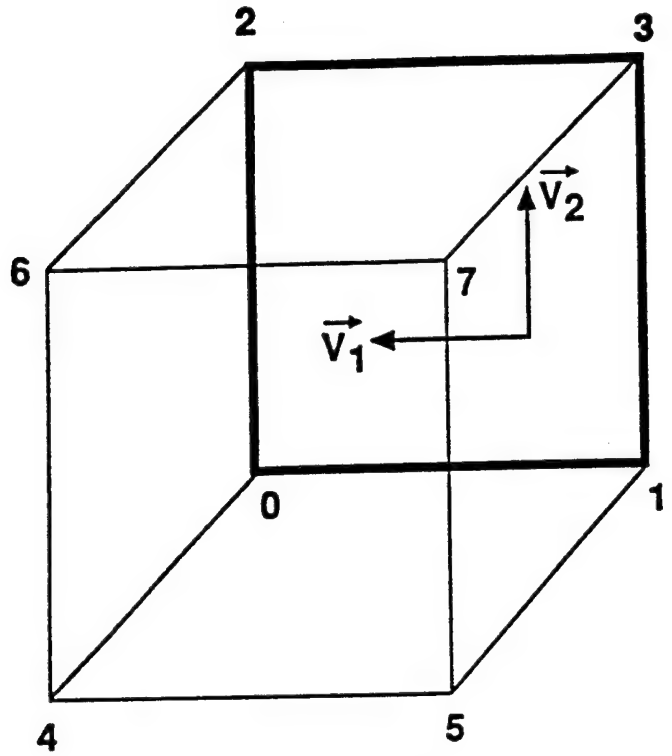
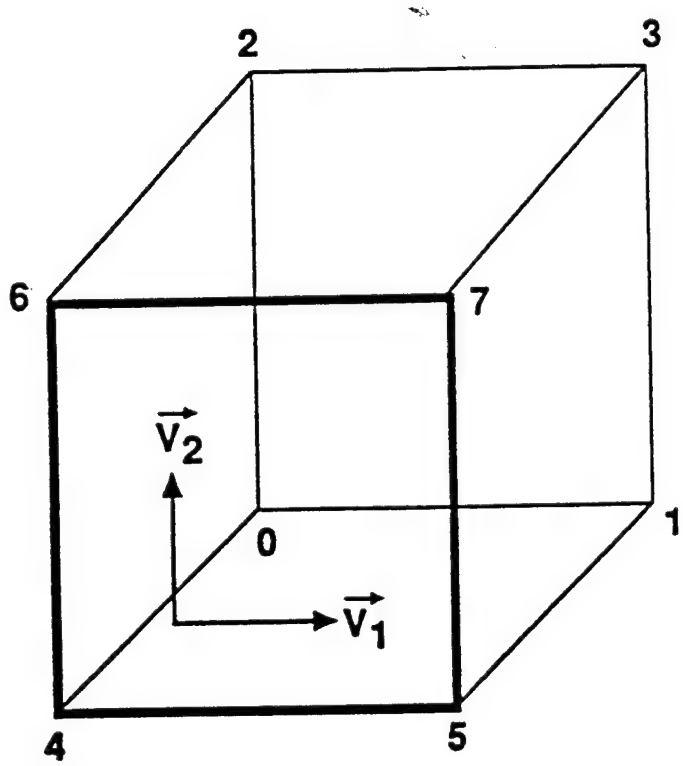


Figure 12.6 Faces and their vertices for tetrahedron



Face 0

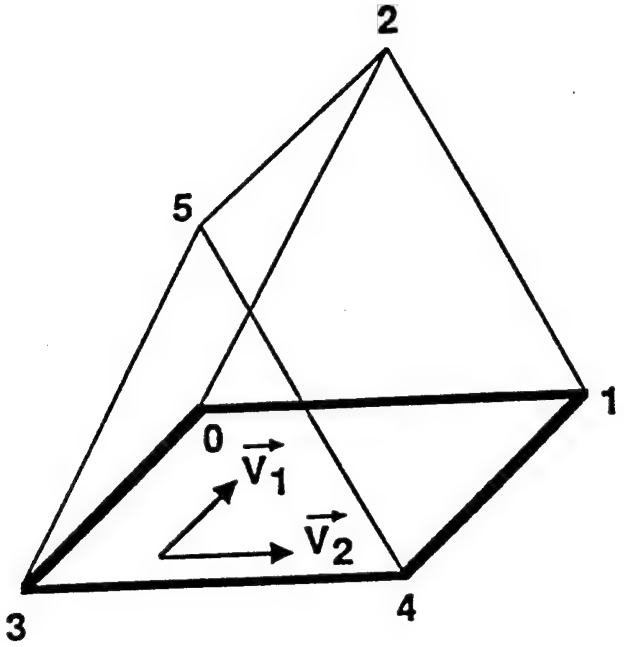
$$T = \begin{bmatrix} -2 & 0 & 0 \\ 0 & 2 & 0 \end{bmatrix}$$



Face 1

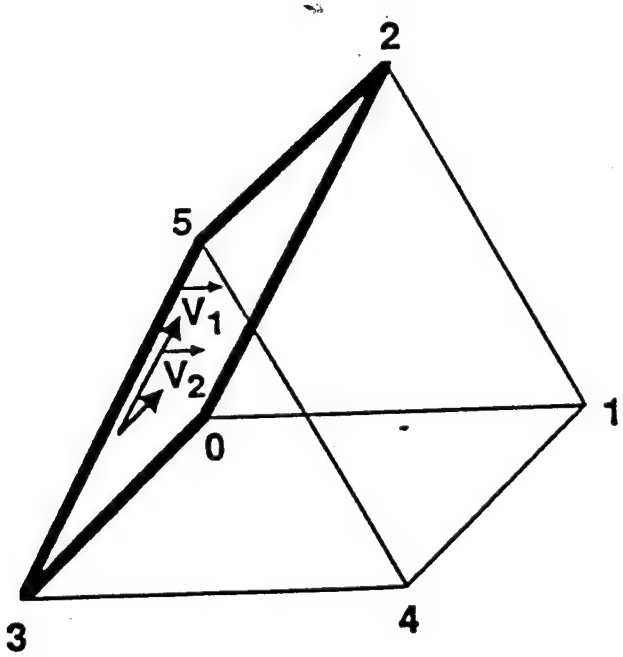
$$T = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \end{bmatrix}$$

Figure 12.7 Faces and their transformation matrix for hexahedron



Face 2

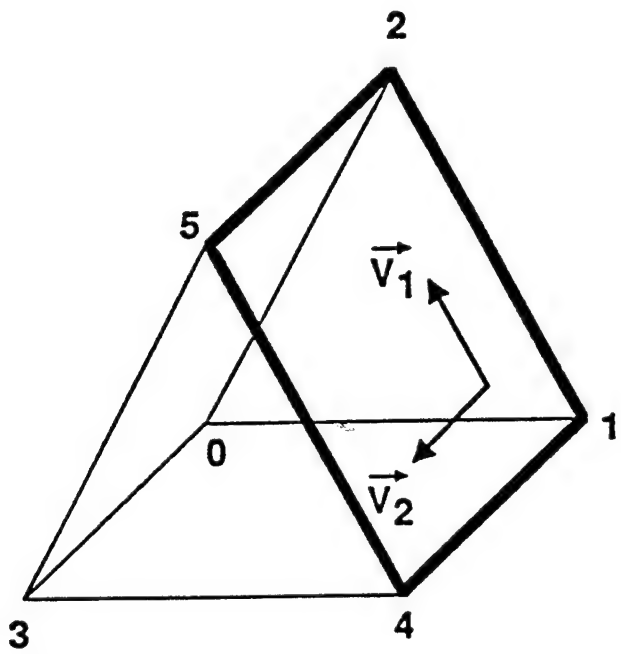
$$T = \begin{bmatrix} 0 & 0 & -2 \\ 2 & 0 & 0 \end{bmatrix}$$



Face 3

$$T = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 0 & -2 \end{bmatrix}$$

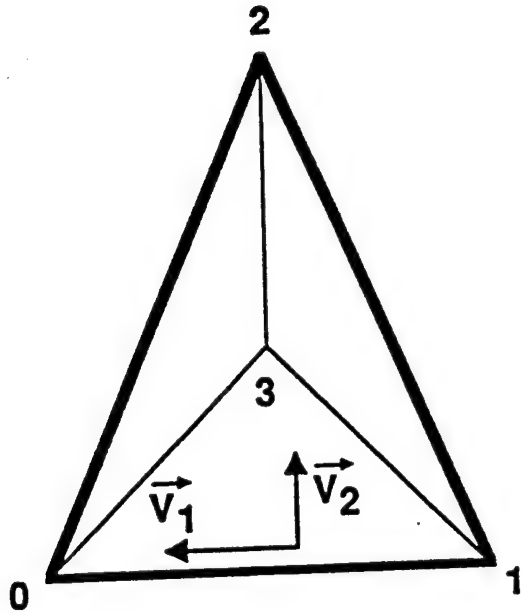
Figure 12.8 Faces and their transformation matrix for triangular prism — continued



Face 4

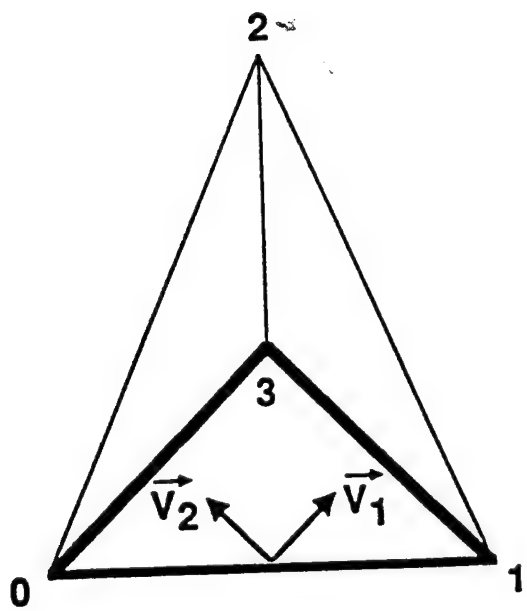
$$T = \begin{bmatrix} -1 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

Figure 12.8 Faces and their transformation matrix for triangular prism — continued



Face 0

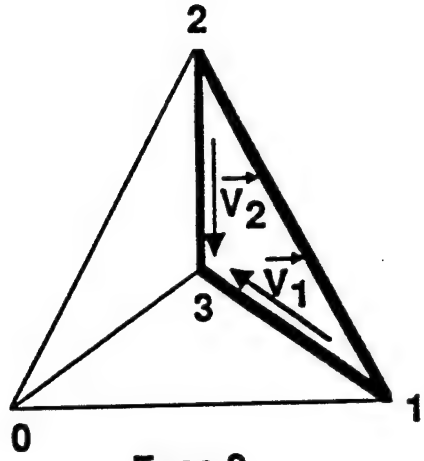
$$T = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \end{bmatrix}$$



Face 1

$$T = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 1 \\ -1 & 1 & 2 \end{bmatrix}$$

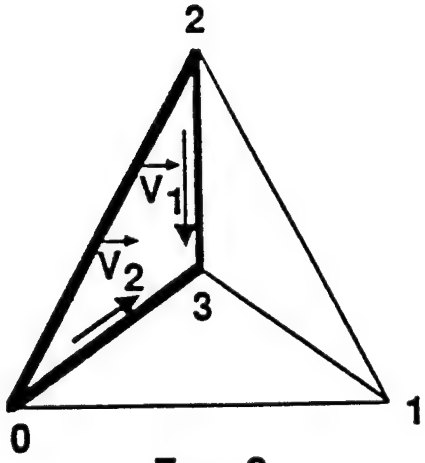
Figure 12.9 Faces and their transformation matrix for tetrahedron



Face 2

Face 2

$$T = \begin{bmatrix} -\frac{1}{2} & \frac{1}{2} & 1 \\ 0 & -1 & 2 \end{bmatrix}$$

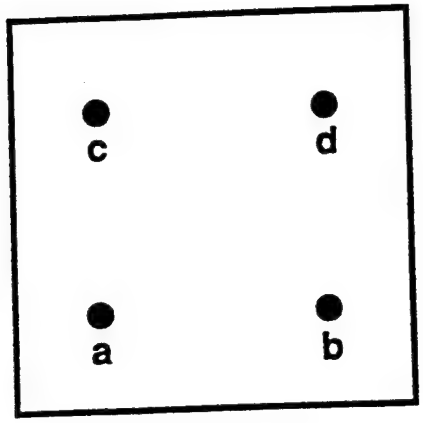


Face 3

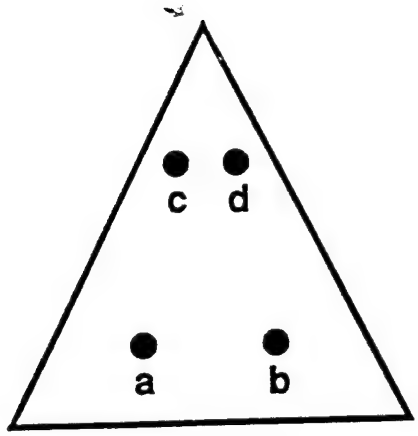
Face 3

$$T = \begin{bmatrix} 0 & -\frac{1}{2} & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

Figure 12.9 Faces and their transformation matrix for tetrahedron — continued

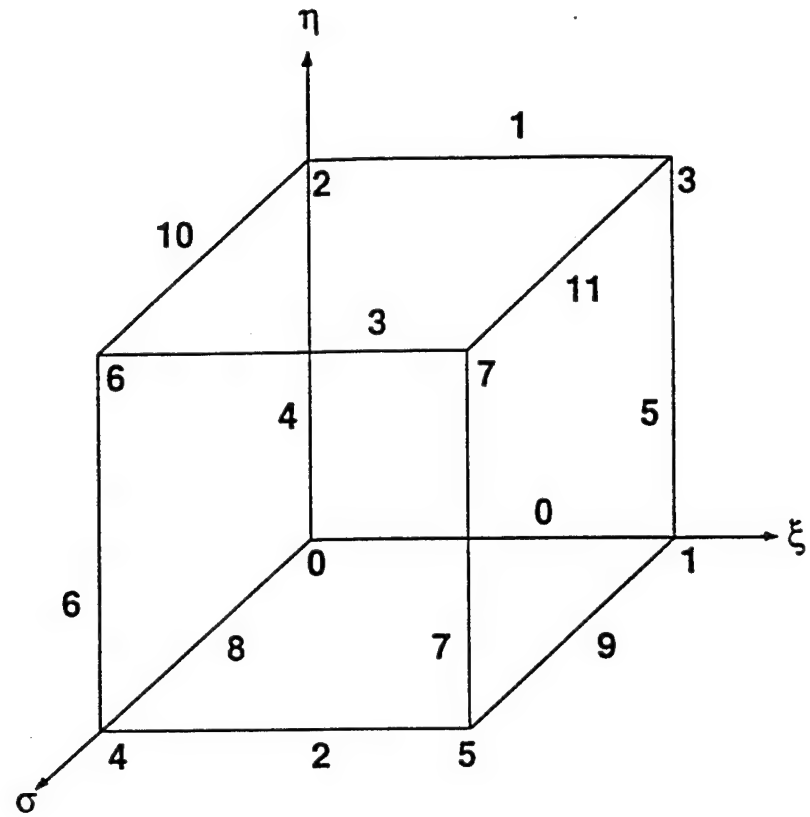


Quadrature points for 4-vertex face



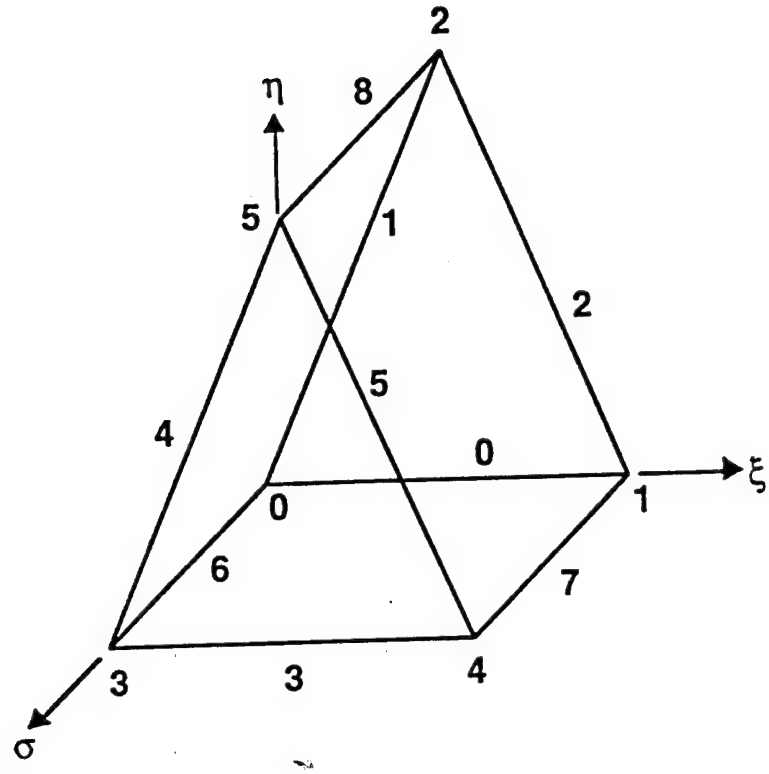
Quadrature points for 3-vertex face

Figure 12.10 Gaussian quadrature points for "square" and "triangular" faces



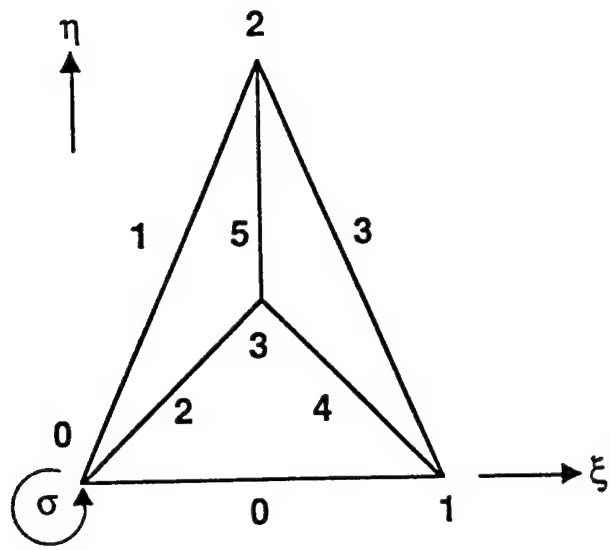
Edge	From Node	To Node
0	0	1
1	2	3
2	4	5
3	6	7
4	0	2
5	1	3
6	4	6
7	5	7
8	0	4
9	1	5
10	2	6
11	3	7

Figure 12.11 Vertex nodes and cell edges for hexahedron



Edge	From Node	To Node
0	0	1
1	0	2
2	1	2
3	3	4
4	3	5
5	4	5
6	0	3
7	1	4
8	2	5

Figure 12.12 Vertex nodes and cell edges for triangular prism



Edge	From Node	To Node
0	0	1
1	0	2
2	0	3
3	1	2
4	1	3
5	2	3

Figure 12.13 Vertex nodes and cell edges for tetrahedron

13.0 MESH GENERATION

The computer code known as TRIM3D in Phase I was renamed UNIVG in Phase II. This naming convention is used to conveniently refer to the UNIVERSE-series suite of software. The unstructured surface grid, viscous triangular prism near-surface grid and setup of boundary conditions are generated interactively in UNISG. The grid generator for tetrahedral volume cells is UNIVG. The flow solver is UNIV. The postprocessing mode of UNIV is referred to as UNIVP.

The computer codes UNISG and UNIVG, based on the two layer method to generate 3-dimensional unstructured grids for viscous and inviscid flows have been developed at Rockwell Science Center. The boundary of the computational domain is divided into several patches. The geometry of each patch is described by specifying a sufficient number of non-intersecting lines on the patch. Each line, in turn, is described in terms of a sufficient number of points on the line (Fig. 13.1). The patch information can also be provided in the form of IGES NURBS surfaces. From this information a parametric representation (s, t) of the surface is developed where s and t are the local running coordinates in the plane of the surface. For convenience, the parameters s and t are chosen to take values between 0 and 1, where $s = 0$, $s = 1$, $t = 0$ and $t = 1$ form the segmented boundary of the patch. Nodes are then distributed on these four boundary lines satisfying the user-specified clustering requirements. In the code, the first and last lines of input patch information correspond to t=0 and t=1 respectively. Presently, the code requires that the number of nodes on at least any one pair of opposite sides ($s = 0$ and $s = 1$ or $t = 0$ and $t = 1$) be equal. This restriction will be removed in the future.

Internal lines are generated by connecting corresponding points on a pair of opposite sides. Each internal line is then divided into line segments. The line segment length at the ends of an internal line is determined by the average of the length of the line segments on the boundary of the patch that intersects the given internal line. The length of the remaining line segments on an internal line is obtained by smoothly blending the lengths of line segments at the ends. Triangular elements are then generated by connecting the line segments according to the recipe described in Fig. 13.2.

The next step is to generate the boundary that separates the viscous and the inviscid regions. For this purpose points are generated along each patch boundary at a specified distance along the normal to the patch. These points are connected to form the boundary of the viscous region. This boundary is then smoothed using an elliptic smoothing procedure and transfinite interpolation is used to generate a smooth surface that is enclosed by it. Projection of triangles on the patch surface onto the newly generated smooth surface is performed. Corresponding triangles on the two surfaces are connected to form triangular prisms. These prisms are first broken into smaller prisms and then divided to form tetrahedra. This process is carried out in such a way that the grid clustering required to resolve small scales near the body surface is achieved. All the connections between patches, direction of volume cell generation, and grid clustering information are interactively setup in UNISG and automatically output in a form compatible directly with UNIVG.

In UNIVG, the "advancing front" method (Ref. 45) is used to fill up the computational domain with tetrahedra starting from triangular elements on the viscous boundary. This technique requires specification of an initial surface divided into triangular elements. The boundary of the computational domain is used as the intial surface of the advancing front. The smallest element on the surface is replaced by the three sides of a new tetrahedron constructed with that element as the base. The fourth node of the tetrahedron is obtained either from one of the existing nodes or by creating a new one (Fig. 13.3).

Grid generation in the inviscid region consists of the following steps:

(1) Define the boundary patches (including body surface and outer boundary).

(2) Triangular elements are then generated on each patch based on user supplied information regarding the number of points and mesh streching.

(3) These triangular elements are used as the initial surface of the advancing front.

(4) Tetrahedral cells are then constructed with the triangular elements as bases. In order to avoid large cells intersecting with small cells, the smallest triangular element from the list of faces is always used.

(5) Select or create a point to form a tetrahedral cell from the base triangular element (ABC, in Fig. 13.4).

a.) The point D can belong to a neighboring triangular element (ACD) as long as it can generate a tetrahedron satisfying certain built-in constraints. (The angles between the base triangular element ABC and newly formed triangles ABD, BCD, CAD should be less than 135 degrees and larger than 10 degrees.) See Fig. 13.4a (top figure).

b.) The point D can be selected from an existing grid or from a list of points supplied for the purpose. See Fig. 13.4b (middle figure).

c.) Otherwise, a new point D located on the line which is normal to the base triangular element at its center can be created to form the new tetrahedron. See Fig. 13.4c (lower figure).

(6) Check whether the newly formed cell intersects with any already existing cells.

(7) If no suitable point can be found, some old cells have to be removed,and repeat step (5).

(8) The new faces ABD and BCD from (5-a) or ABD, BCD, CAD from (5-b) and (5-c), the new cell ABCD and new point D from (5-c) are all added to their respective lists (face, cell and point).

(9) Delete the old face (ABC) from the list.

(10) Check if there are any triangular elements remaining in the front . If yes, go back to step 4.

The two most important aspects of UNIVG are 1) data structure and 2) the logic to check whether any two tetrahedra intersect with each other. These aspects are disscussed in the following section.

13.1 Data Structure

In order to find the smallest triangular element, points in the neighborhood and

neighboring triangular elements efficiently, the data structure for the faces, and the grid points is arranged as follows:

13.1.1 List for The Front Face

A binary tree structure is used to form the front face list. The ordering of the tree is arranged such that the area of the father face is smaller than the face of the two sons. Figure 13.5 shows a binary tree of this form. The area of each face and its position are also shown in this figure. It is noted that the position of the two sons are located at $Ison1 = Ifather \times 2$ and $Ison2 = Ifather \times 2 + 1$, respectively. For example, Face D located in position 4 has two sons located at position 8 (H) and 9 (I). A face should be added or deleted without altering the binary tree structure. The ideas of the heap-sort and heap-search algorithms (Ref. 46) are used in UNIVG.

a) Adding a new face to the face list:

A new face is always added first at the end of the tree. Then, the internal order of the binary tree is rearranged by comparing the face areas of fathers and their sons. This procedure can be described as follows:

1) A face J with area 4.0 is to be added to a binary tree in Fig. 13.6.

2) Place J at the end of heap list (10 in this case).

3) Find the father's position of 10 by using $Ifather = Ison/2$. Therefore the father's position is 5 and the face is E in this case.

4) Compare the area of faces J and E.

5) If the area of J is less than that of face E, interchange the position of father and son (see Fig. 13.6a).

6) Unless J is at position 1 (top of the list) go back to (3).

b) Delete a face from the front face list:

A face can be removed from any position from the tree. Then, the internal order of the tree is re-established by comparing the area of the face of the father and son.

1) Find the position from which the face is to be removed from the tree. For example, we can consider face A at position 1 in Fig. 13.6b.

2) Place the face stored at the end of the tree at this position (1 for this case).

3) Find the two sons location by using $Ison1 = Ifather \times 2$ and $Ison2 = Ifather \times 2 + 1$.

4) Determine if the father and son's position should be changed by checking the area of the father face and two sons' faces.

if

$(Area(father) < min(Area(son1), Area(son2))$: no change

else

change father's position with the smaller area son's position.

5) Repeat step 4 until no change is reqired (Fig. 13.7).

In this binary tree, the face with smallest area will always remain at the top of the list and can be used as the next surface to be removed.

13.1.2 List for Points in the Neighborhood

An octree data structure is used to efficiently locate points in a neighborhood. The fist octant is determined from the boundary of the computational domain. At most, eight points are stored in each octant. If a ninth point falls into an octant, then it is subdivided into eight smaller octants. This procedure is continued until an octant with vacant storage is found. Figure 13.8 illustrates this process. A new point I is added and it falls into octant 1 which already contains eight points A,B,C,D,E,F,G,H. Therefore octant 1 will be divided into eight smaller octants (octants 2-9). The newly added point I with old points D,E,F are relocated in octant 2. In the UNIVG code ,an array $IOCTR(11, MXOCT)$ is defined to store the points. The variable $MXOCT$ is the maximum number of octants allowed. For each octant IOC, the following

information is stored.

$IOCTR(11, IOC) = -1$: the octant is full

$IOCTR(11, IOC) = 0$: the octant is empty

$IOCTR(11, IOC) > 0$: the number of points stored in this octant

$IOCTR(1:8, IOC)$: point numbers are stored here if $IOCTR(11, IOC) > 0$

The maximum and minimum x, y, z are also stored for each octant. By using this octree, the neighboring points within some specific distance of a given point (x, y, z) can easily be determined.

13.1.3 Linked List Between Point and Faces

In an unstructured grid generation code, it is important to determine the faces that surround a given point. In order to do that, an address pointer array "$LPION(IPONT)$"
for each grid point IG is first allocated. The faces surrounding point "IG" are saved in an array $LFAPO(8, IG)$. The procedure to find faces surrounding a given point IG is described below:

1) Find the address related to the given point IG.

$$IADRES = LPION(IG)$$

2) $LFAPO(8, IADRES) > 0$ defines number of stored faces.
   $LFAPO(8, IADRES) < 0$ implies that next face number surrounding the grid point IG is continued at the address $abs(LFAPO(8, IADRES))$.

3) $LFAPO(1:7, IADRES) = 0$ defines an empty location
   $LFAPO(1:7, IADRES) > 0$ denotes face number for face which surrounds the point IG. This method is illustrated in Fig. 13.9.

13.2 Check for Intersecting Faces

In the process of generating cells, no two faces must intersect each other. This condition can be satisfied only if no side of either face intersects the other face (Figure 13.10). A total of six conditions (One each for the six sides that make up two triangles.) have to be checked to make sure these two faces do not cross each other. With the notation given in the figure, determine whether a side $\vec{g}_3$ (connecting $\vec{x}_4$ and $\vec{x}_5$) intersects the face $\vec{g}_1\vec{g}_2$ $(\vec{x}_1, \vec{x}_2, \vec{x}_3)$. The intersection point of line $(\vec{x}_4\vec{x}_5)$ with the plane defined by $\vec{x}_1$, $\vec{x}_2$ and $\vec{x}_3$ is given by

$$\vec{x}_I = \vec{x}_4 + \frac{(\vec{g}_1 \times \vec{g}_2) \cdot (\vec{x}_1 - \vec{x}_4)}{(\vec{g}_1 \times \vec{g}_2) \cdot \vec{g}_3} \vec{g}_3$$

The segment $(\vec{x}_4\vec{x}_5)$ intersects the triangle $(\vec{x}_1, \vec{x}_2, \vec{x}_3)$ if $\vec{x}_I$ belongs to the triangle itself. This procedure leads to three conditions for each triangle. If all six conditions are satisfied, then the two faces do not cross.

13.3 Grid generation for viscous flow

1) Follow the first three steps of the inviscid grid generation.

2) Generate the boundary of the viscous region by moving the patch boundary along its normal a specified distance "d" (Fig. 13.11). Since the normal at a point is not uniquely specified, an average of the normals to the triangles that meet at the given point is determined by taking the average of the normals to the two adjacent triangles that differ the maximum. A point is then generated at a given distance along this normal.

3) Employ an elliptic smoother to smooth the boundary.

4) A smooth surface enclosed by this boundary is generated using transfinite interpolation. This process involves creating a map between the local surface coordinates (s, t) and the cartesian coordinates (x, y, z).

5) Project the triangles on the body surface patch onto the surface generated in step 4. This is done by locating points on the new surface at the same (s, t)

locations as the (s,t) coordinates of the vertices of the triangles on the body surface patch.

6) At this stage the newly generated points for all the body surface patches are checked for positive distance from the body surface (Fig. 13.12). Points with negative distance from the body surface are corrected by a trial and error procedure.

7) Corresponding triangles on the two surfaces are connected to form triangular prisms. Any occurance of twisted prisms is rectified by moving the points involved using an elliptic smoother. The procedure for checking whether a triangular prism is twisted or not is given in Figure 13.13.

8) The triangular prisms are broken into smaller prisms to ensure that the flow in the boundary layer region is properly resolved (Fig. 13.14a).

9) The smaller tringular prisms are divided into tetrahedra (Fig. 13.14b).

Figure 13.15 shows the body surface, smooth surface and projected triangles of the space shuttle nose region.

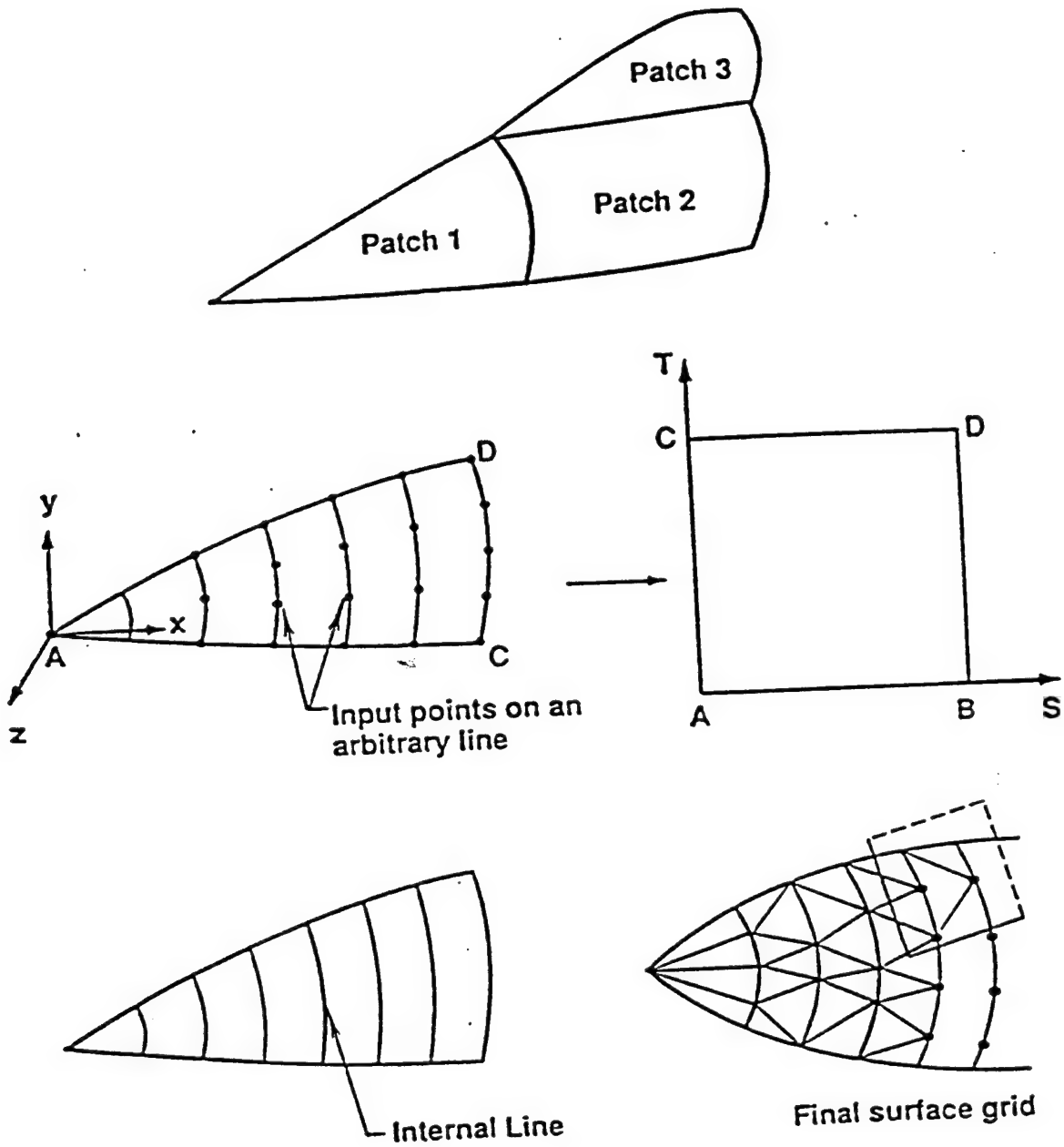
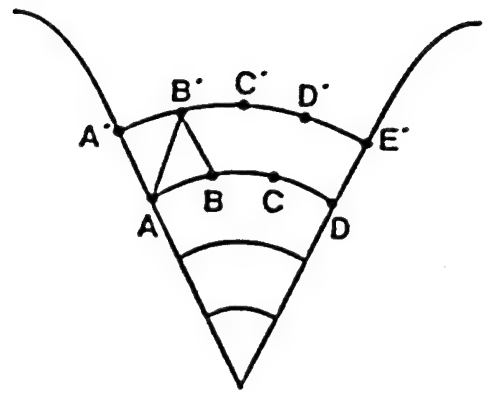
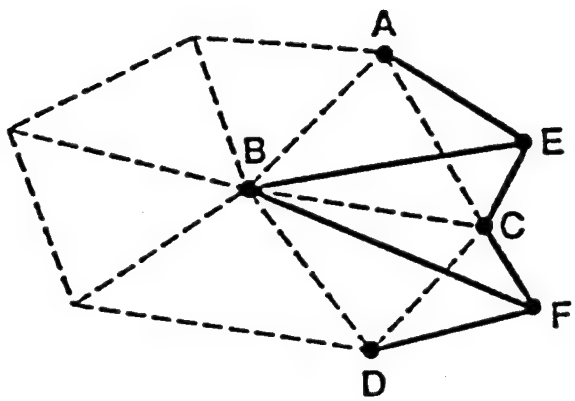


Figure 13.1 Surface grid generation



1) start with (A,B).
2) There are 2 choices:
   a) A can be connected to B'
   b) B can be connected to A'
3) compute lengths of AB' and BA' and choose the shorter one; for e.g., AB'.
4) since B is not connected yet, continue the process with (A,B).
5) 2 choices :
   a) A can be connected to C'
   b) B can be connected to B'
6) compute lengths of AC' and BB' and choose the shorter one; for e.g., BB'.
7) since both A and B are now connected, continue the process with (B,C).

Figure 13.2 Recipe for generating triangular elements



- - - - : initial front surface divided into tringular elements.

ABC & CBD : two of the initial surface elements.

E & F : newly generated nodes.

ABE , BCE , CAE : new elements that replace ABC.

CBF , BDF , DCF : new elements that replace CBD.

BFE & FCE : new elements generated by connecting two existing nodes E and F. They replace the elements BCE and CBF.

Figure 13.3 Advancing front technique

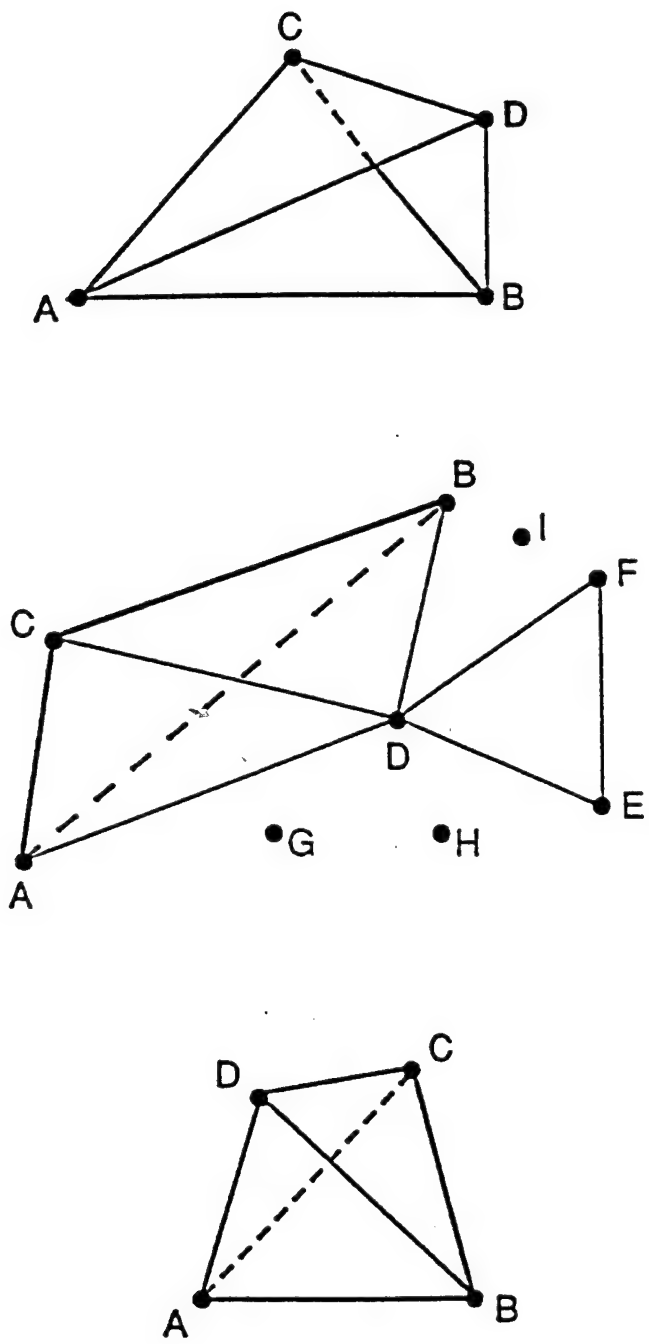


Figure 13.4 Construction of new element

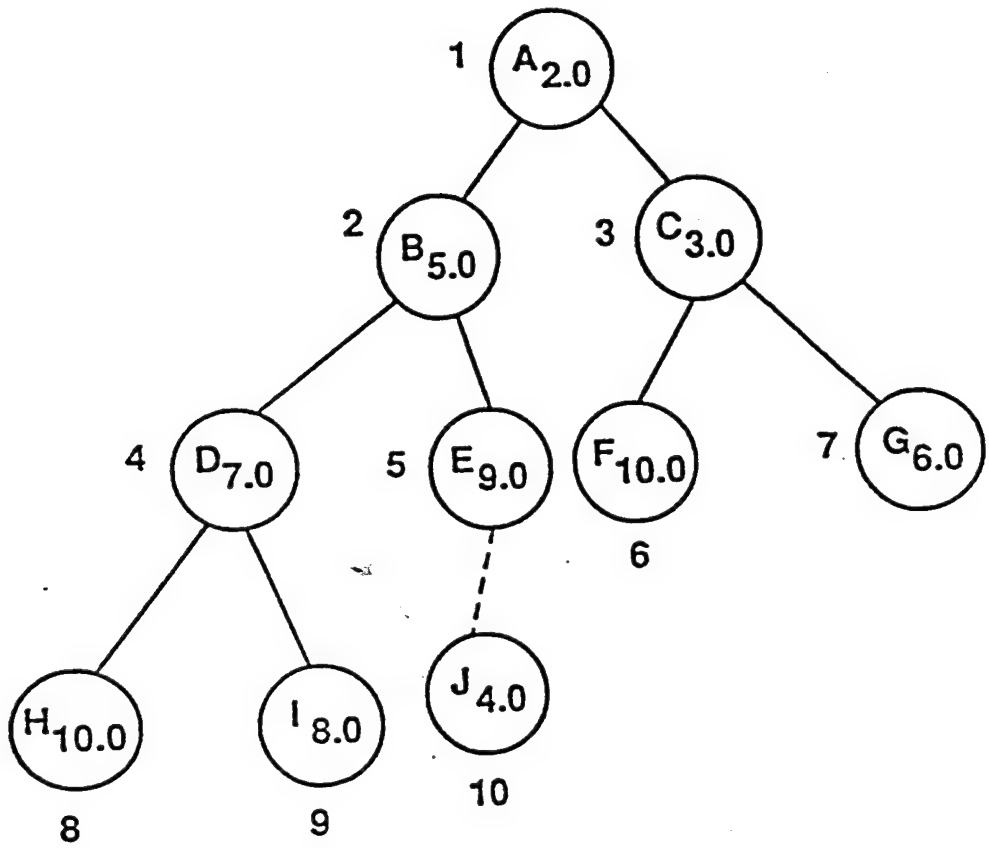


Figure 13.5 Binary tree

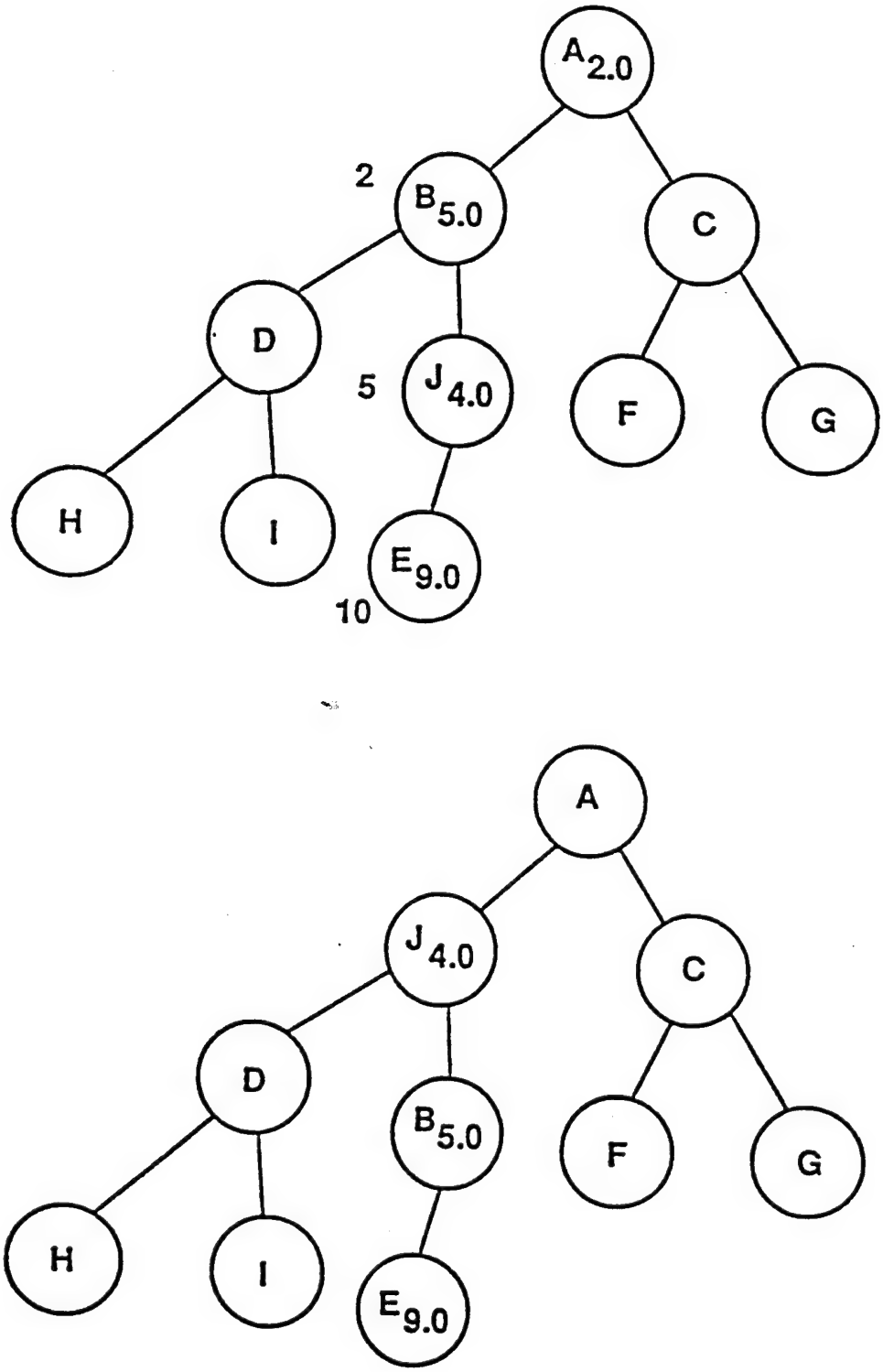


Figure 13.6 Addition of new face to binary tree

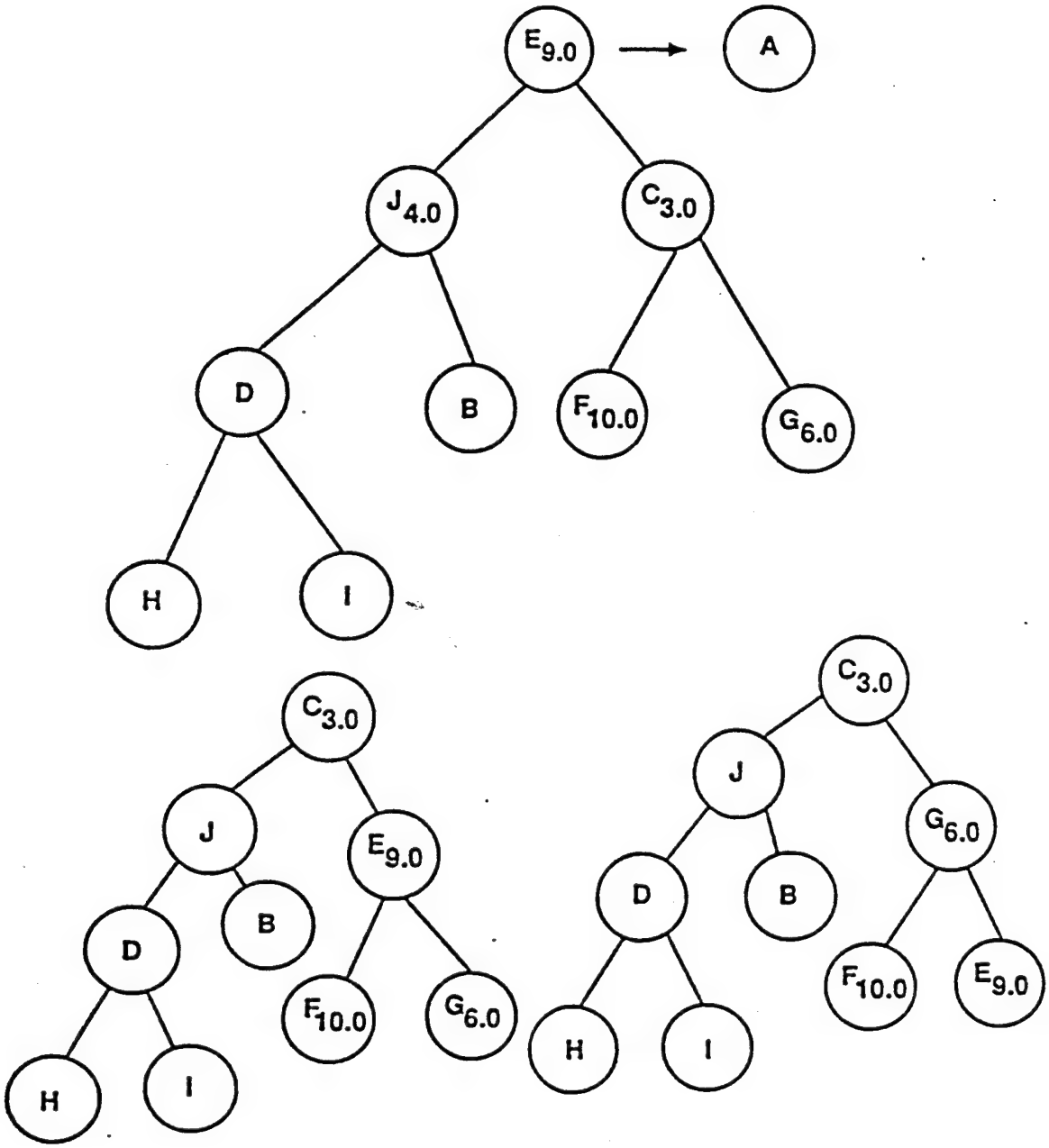


Figure 13.7 Removing a face from binary tree

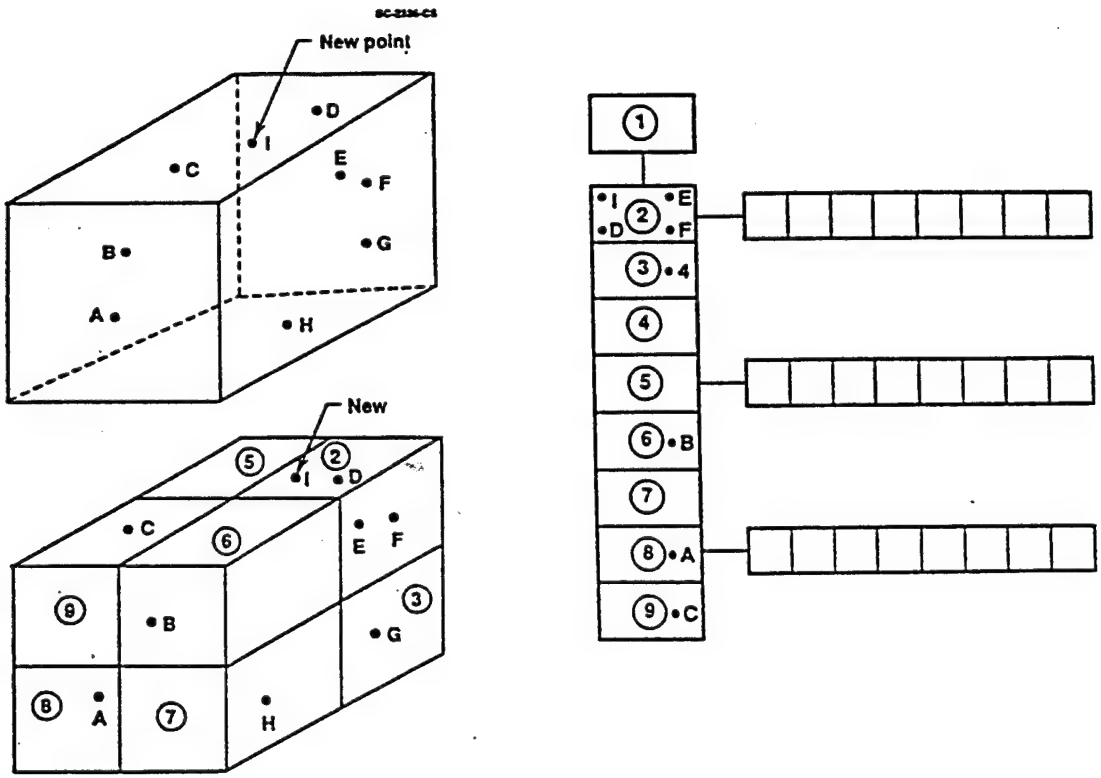


Figure 13.8 Construction of octree

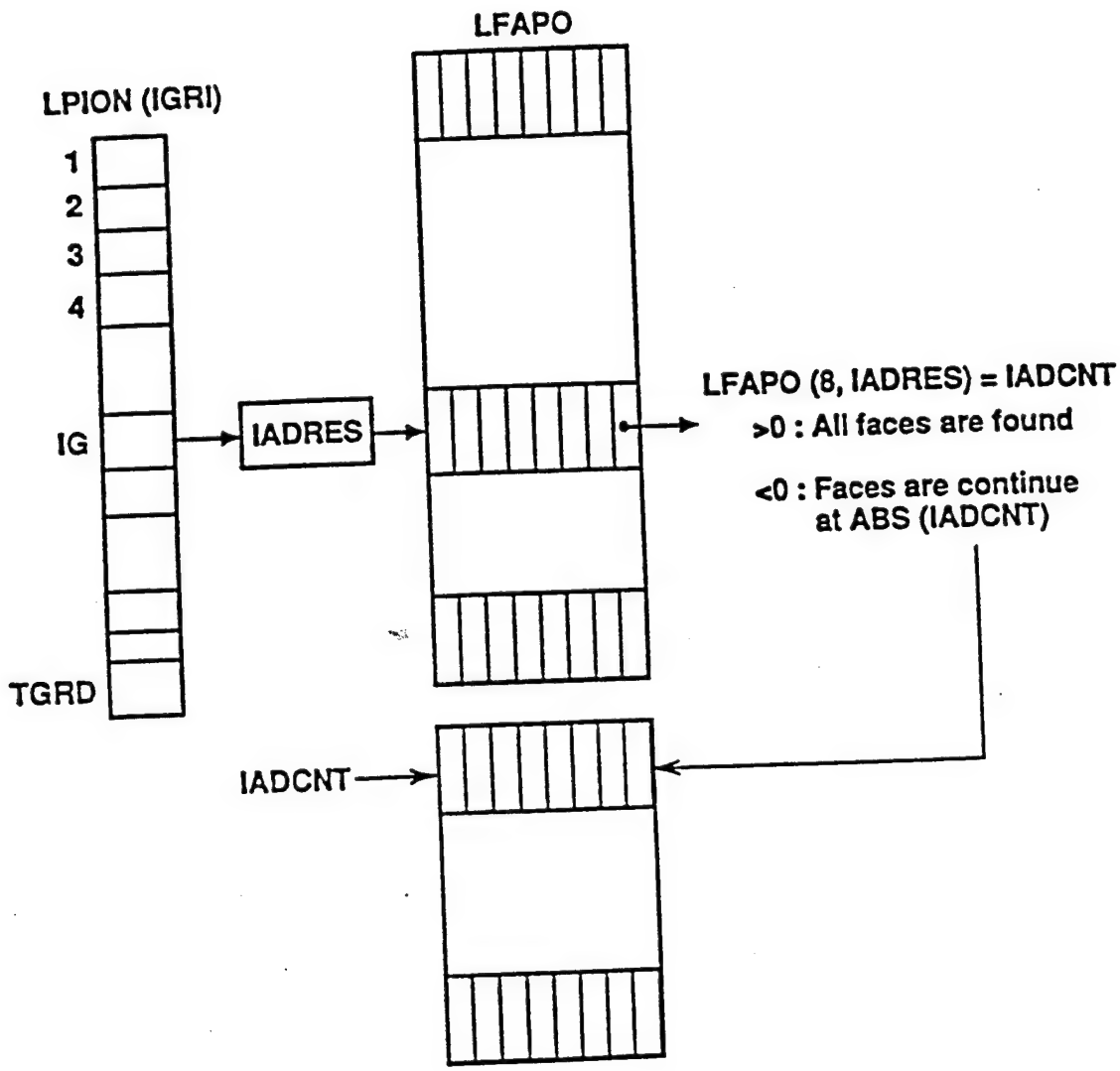


Figure 13.9 Linked list between point and faces

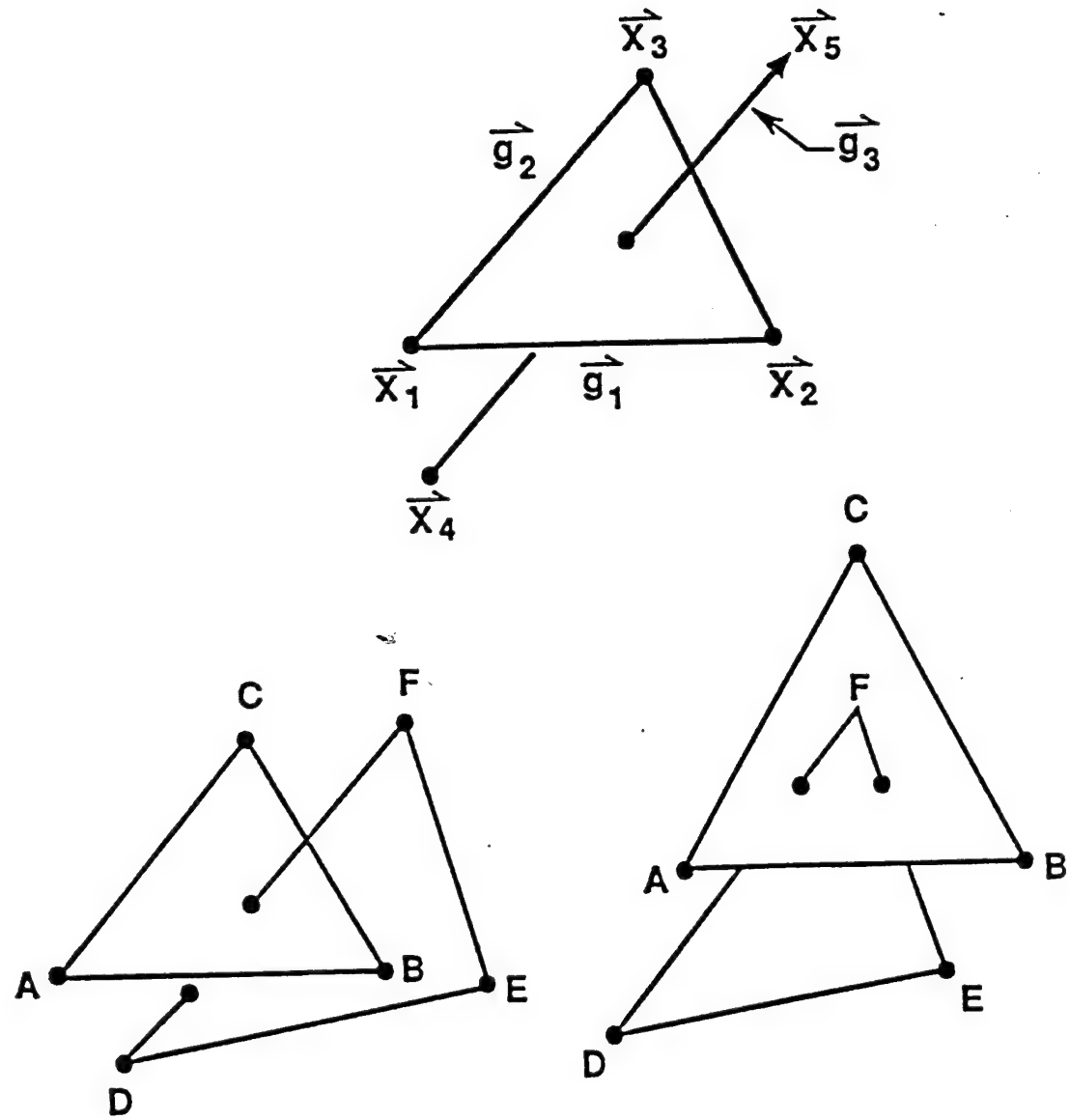


Figure 13.10 Intersection of two faces

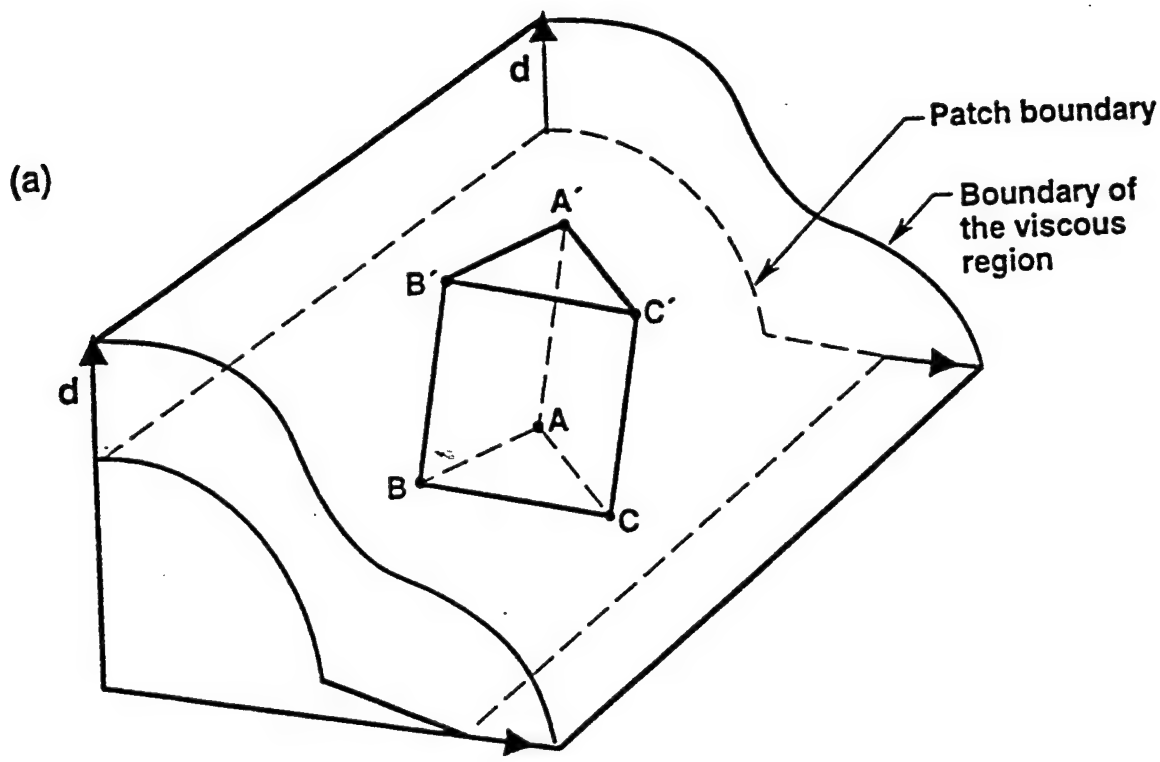
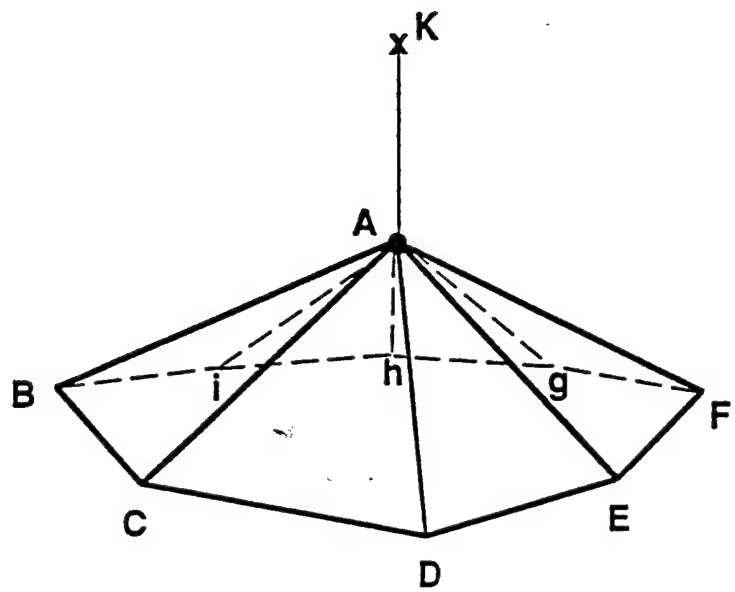
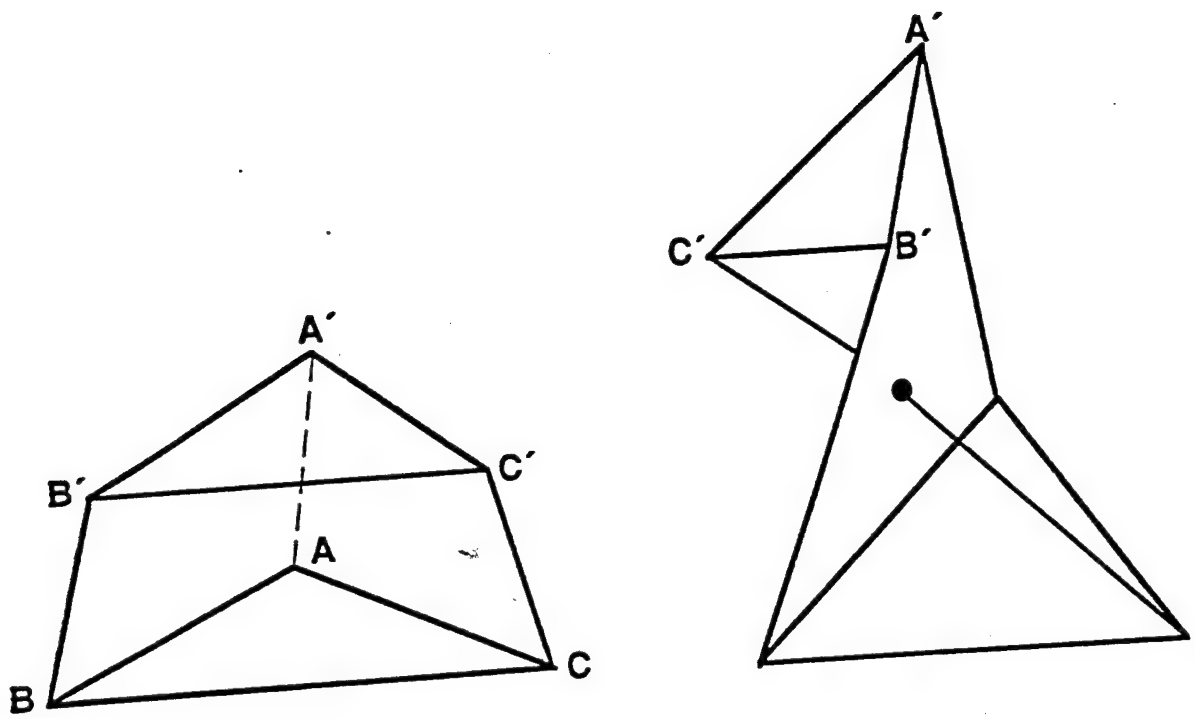


Figure 13.11 Boundary of the viscous region



FOR ΔABC, $(\overrightarrow{AB} \times \overrightarrow{AC}) \cdot \overrightarrow{AK} > 0$ has to be satisfied.
Same as all other triangle ACD, ADE..etc.

Figure 13.12 Criterion for accepting newly generated point A



$(\overrightarrow{A'B'} \times \overrightarrow{A'C'}) \cdot \overrightarrow{A'A}$ has to be negative

Figure 13.13 Criterion for accepting newly generated triangular prism

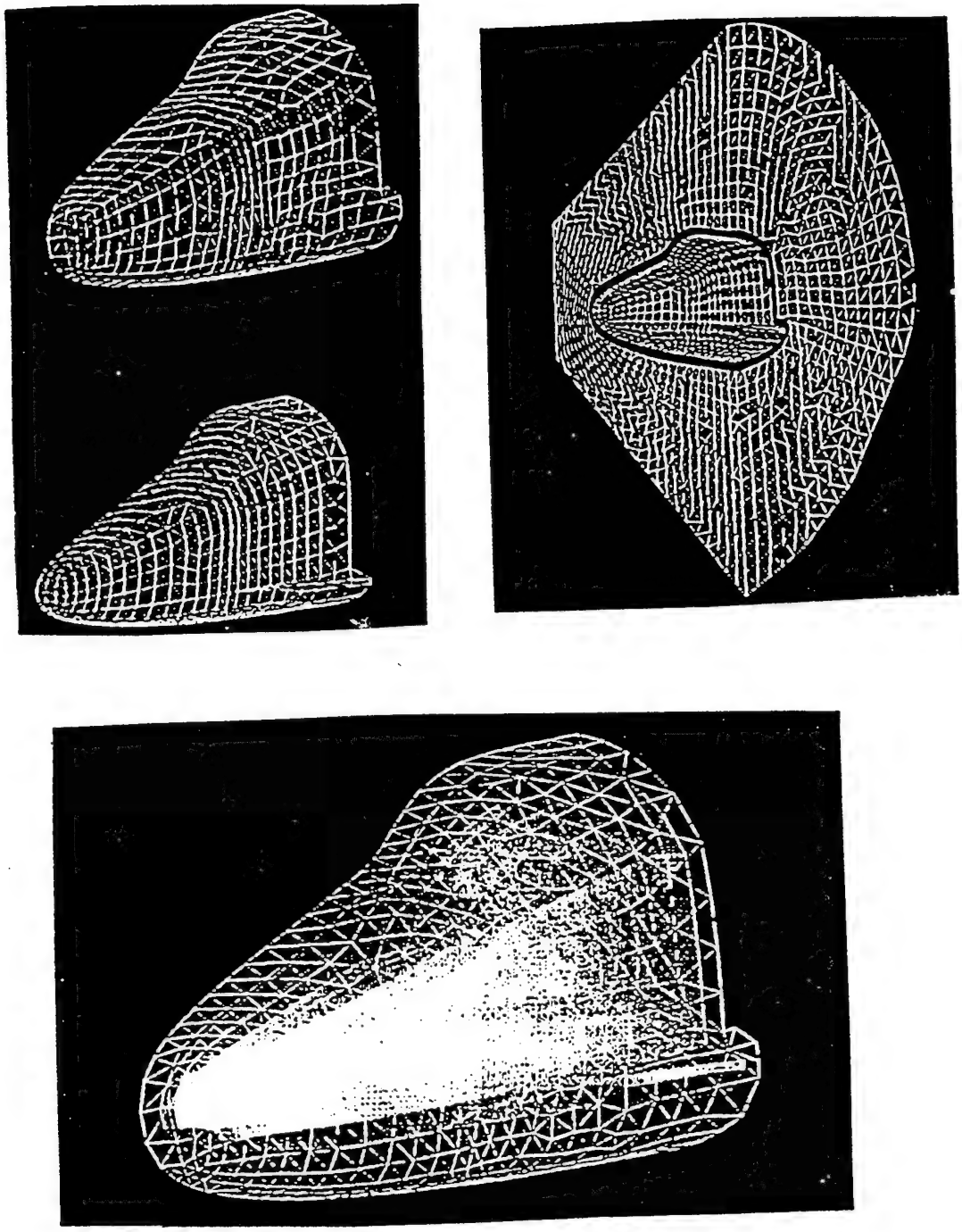


Figure 13.15 Viscous grid for the Space Shuttle nose region

14.0 TOPOLOGY TREATMENT

This section describes various topology issues related to connectivity of nodes into cells, cells common to a node, cells with a common face, how cells may be subdivided, how links may be removed, etc.

14.1 Node Locations and Nodes of Cell

The computational unstructured mesh is defined by 1) specifying the set of nodes and their locations (x, y, z coordinates and optionally their nodal velocities $\dot{x}, \dot{y}, \dot{z}$), and 2) specifying the set of nodes that comprise each cell. These node numbers are called the **global** node numbers. Each cell has an ordered collection of nodes numbered locally. When the global nodes of each cell are provided, they must be given in the same order as the local node numbers. Therefore, given any local node number, the corresponding global node number is easily determined.

14.2 Elimination of Redundant Nodes and Cells

Let us make a set of all nodes that are part of at least one cell in the total collection of cells given. This may be a subset of the collection of nodes for which the locations have been provided. The extra nodes in the larger set are those nodes not needed to define any cell. These nodes (and their locations) may have been generated during the mesh generation process but were not used because their inclusion would have resulted in cells that were somehow unsatisfactory. The ability to remove redundant nodes is provided for in the unstructured-grid UNIVERSE-series formulation.

During the process of removing links, a topic to be covered later in this section, some cells may become degenerate. This may happen in two ways: 1) the number of distinct global vertex node numbers is less than four; 2) the number of distinct vertex node locations is less than four. Case 1 deals with a "logically" degenerate cell and case 2 with a "physically" degenerate cell. Removing these cells become redundant for the purpose of computing the dependent variables. Removing such redundant cells corresponding to case 1 is available in UNIVERSE-series unstructured-grid formulations as part of various "grid editing" options.

14.3 Cells of Given Node

Given the nodes of each cell, the inverse map can be constructed. This provides knowledge of which cells include the given node. While each cell has the same number of nodes (assuming the same cell type for every cell), the number of cells that include a given node varies with each node. We have verified for linear elements (only vertex nodes) that the sum of cells associated with each node is equal to the number of cells times the number of nodes per cell which is a somewhat surprizing statement at first glance.

14.4 Common Faces

Each face is made up of a specific set of local node numbers, each of which is mapped onto a global node number. Using the knowledge of nodes of a cell and cells of a node, the cells that share a common face can easily be identified.

Faces are numbered sequentially with cell number. The first six faces belong to the first cell, the next six faces belong to the second cell, etc., for the hexahedral cell type. Therefore, face numbers are related directly to cell numbers. A face shared by two cells sports two face numbers, one that identifies it as part of the first cell and another that identifies it as part of the second.

A given face of a given cell has a number of vertex nodes. Each vertex node is associated with many cells. Out of these, one cell is the cell pointed to by the face number being considered. There is not more than one more cell which is common to all nodes of the face. In this fashion, the cells that share a given face can be determined. It then becomes a relatively trivial matter to identify which face of the second cell is identical to the face being considered. Thus the face number corresponding to the neighbor cell can be identified, given the face number of a particular cell. Of course, at a boundary, a face can only be part of one cell, the interior cell.

14.5 Cell Types

A major enhancement added during Phase III was the ability to deal with different cell types in the mesh. In fact any cell could be of any of the types allowed by

UNIV (tetrahedral, triangular prism or hexahedral). The user specifies all the types that are expected to be encountered in the mesh. At the time of reading in the mesh, UNIV sets up an cell type identifier for each cell. Each cell type has an associated C structure that completely describes the local book-keeping information for that type. Whenever necessary, a cell refers to its type. This, in turn, points to all attributes of that type.

14.6 Cell Division

For various reasons, a cell may have to be divided into two cells. In Phase I, cell division capability had been developed for the triangular prism cell type for use in two-dimensional inviscid store-tracking studies. Figure 14.1 displays a situation where one face of a triangular prism cell has been divided. If the face is common to two cells, this leads to both cells being subdivided into two cells each.

The following book-keeping details must be kept track of:

1. New nodes are introduced (with global numbers greater than the existing maximum).
2. New cells must be constructed (with cell numbers assigned to be greater than the existing maximum values).
3. Two existing cells must be redefined to account for their being made of different nodes after subdivision.
4. "Cells of node" must be recomputed in the vicinity of the subdivided cells.
5. "Face to face" correspondence must also be reestablished locally.

Such a grid "editing" capability has been developed. The cell division capability is currently not being used (at end of Phase III).

14.7 Link Removal

It is sometimes desirable to remove links. In Fig. 14.2, the short link *ab* is an example. The link can be removed by moving one of its nodes towards the other. In

this example, cells 2 and 5 become degenerate and must be removed from consideration. Node b merges with a and therefore a is used to replace b in all cell definitions. Consequently, node b becomes redundant and can be removed from consideration if necessary. The book-keeping steps are given below.

1. Node b for all cells associated with b must be replaced with node a (nodes of cells 3 and 4 must be redefined).

2. Collapsed cells 2 and 5 must be removed from the database.

3. If node b is not removed from database and must be used for some purpose, the location coordinates of b must be modified to coincide with node a's position.

4. "Cells of node" and "face of face" must be recomputed in the local region.

Such a grid editing capability has also been developed. The link removal capability is currently not being used (at the end of Phase III).

14.8 Octree Sort and Search

The octree sort and search procedure applied to spatial data is the three-dimensional equivalent of the binary sort and search for one-dimensional data. We exploit this heavily to search efficiently for a point nearest to a given point, a cell that contains a given point, intersection of an interior surface with cells, association of mesh point location with surface boundary location, etc.

We begin by sorting available mesh points into the octree spatial data structure. Typically, we restrict the maximum number of physically distinct mesh points in any given octree leaf to 8. Duplicate mesh points (different points at the same physical coordinates) are included and do not have to be discarded in our implementation. In the following discussion, "leaves" are terminal branches of the octree. Nontrivial database information is stored only on octree leaves. Octree branches that are not leaves serve only in the role of a "container" pointing to the next level of branching or subdivision. The two basic procedures that can be applied to the octree database are as follows:

(a) Given a target point, find the octree leaf that contains it.

(b) Given a "box" (of given $\Delta x, \Delta y, \Delta z$ dimensions), find all the octree leaves that intersect with it.

1) By making use of (a) and (b), it is easy to determine the mesh point (that is in the octree database) nearest to a given target point (that may or may not be in the octree database). This is used repeatedly in our grid generation method for constructing unstructured grids (with tetrahedral or triangular prism cells).

2) We also construct an auxiliary database from the octree database. This is the list of mesh cells that are associated with each octree leaf. In its crudest form, for each cell its "container" box is constructed and procedure (b) above is used to find all leaves that intersect with this box. The cell under consideration is added to the list of "associated" cells for all these leaves. From this information, the following procedure can be constructed:

(c) Given a target point, find the mesh cell that contains it.

3) This is accomplished by first using (a) to determine the octree leaf that contains the target point and then identifying all the cells that are associated with this leaf. These and only these cells are searched to determine if the target point is contained within. If none of the cells contains the point, the "nearest" cell from this list can be identified as that cell for which the normal distance from the point to any of the cell's faces is the shortest. This normal distance is defined to be positive when the point is outside the cell and negative when it is contained within the cell. This "positive" distance will come in handy in the later section dealing with the idea of the "alternate" boundary condition.

4) Similar to the database in Item 2 above, we also construct (when necessary) an auxiliary database that lists the surface elements (or surface cells) that are associated with each octree leaf. These surface cells are used to describe the geometry of physical or fluid-dynamic entities that are either at the boundary or in the interior of the flow-field (and computational) domain. They may also be at mesh-block boundaries. Using this database, the following two procedures may be defined.

(d) Given a set of surface elements (or even a single one), find the mesh cells that are associated with the same octree leaves that the surface element(s) are associated

with.

(e) Given a set of mesh cells (or even a single one), find the surface cells that are associated with the same octree leaves that the mesh cell(s) are associated with.

5) Procedures (d) and (e) lead to a very efficient way of isolating a few surface elements that correspond to a given mesh cell and can be used to define a method to find the intersection of surface elements and mesh cells:

(f) Given a mesh cell, find the intersection of the edges of this cell and a given surface.

6) Procedure (f) can be used to describe the intersection of the mesh and the surface geometries. This can be used to generate interior or boundary condition specifications. It can also be used for postprocessing applications where it is necessary to construct the solution on an arbitrary cutting plane.

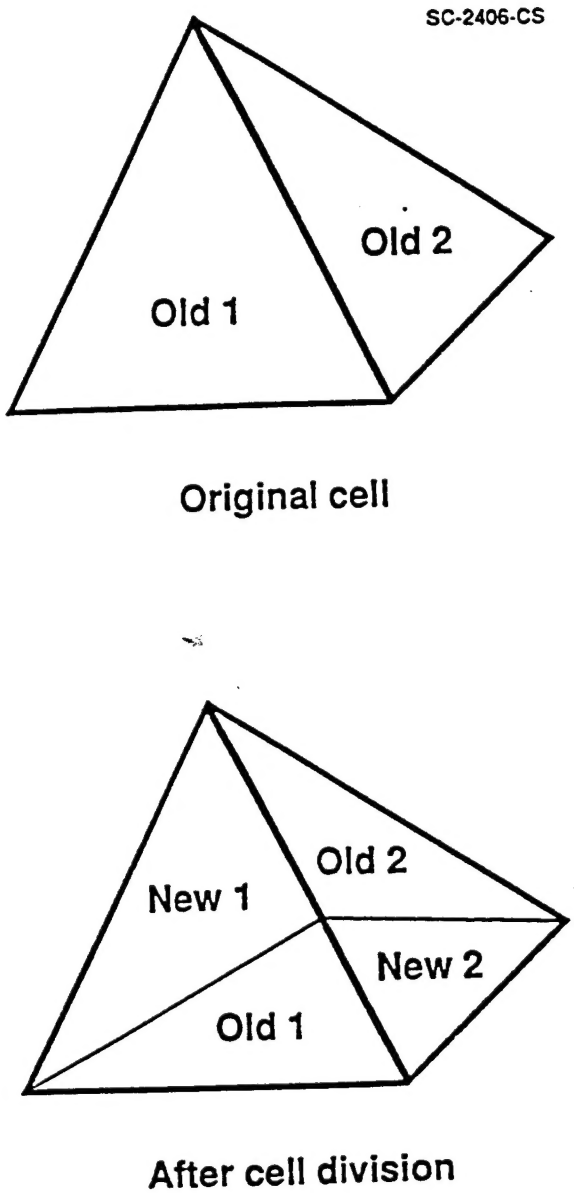


Figure 14.1 Cell division for triangular prisms

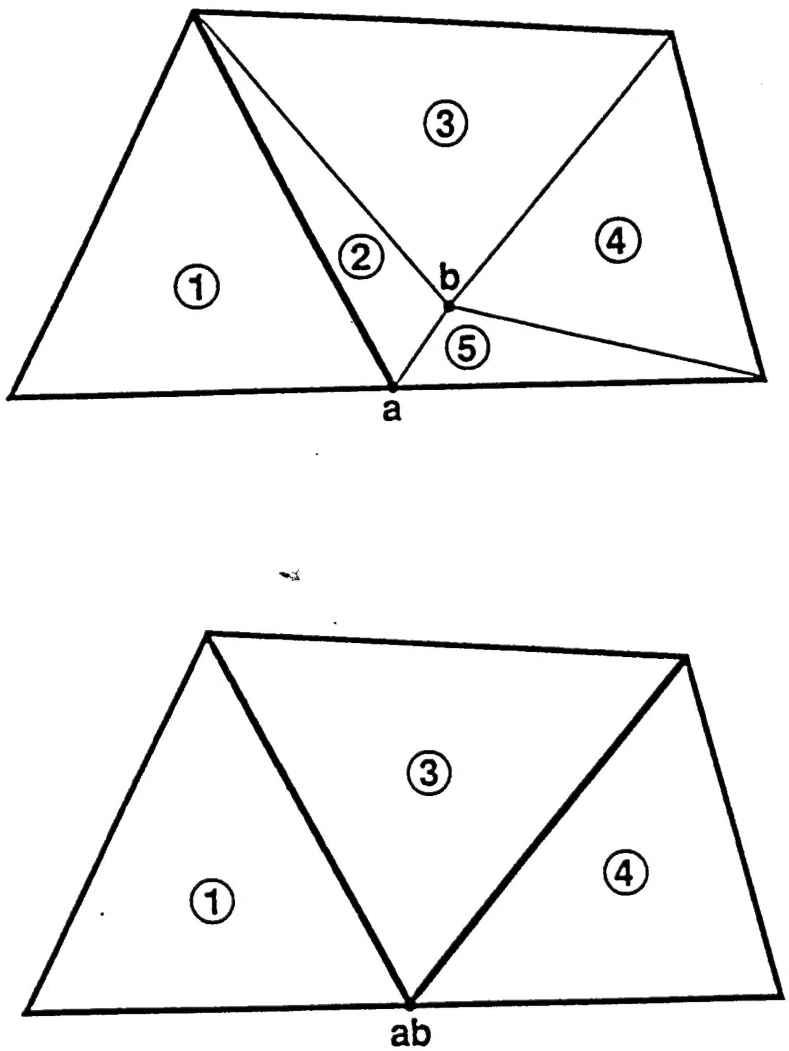


Figure 14.2 Link removal for triangular prisms